

Dhinaharan Nagamalai
David C. Wyld (Eds)

Computer Science & Information Technology

6th International Conference on Computer Science and Information Technology
(CoSIT 2019) February 23-24, 2019, Dubai, UAE



AIRCC Publishing Corporation

Volume Editors

Dhinaharan Nagamalai,
Wireilla Net Solutions, Australia
E-mail: dhinthia@yahoo.com

David C. Wyld,
Southeastern Louisiana University, USA
E-mail: David.Wyld@selu.edu

ISSN: 2231 - 5403
ISBN: 978-1-921987-98-4
DOI : 10.5121/csit.2019.90201- 10.5121/csit.2019.90209

This work is subject to copyright. All rights are reserved, whether whole or part of the material is concerned, specifically the rights of translation, reprinting, re-use of illustrations, recitation, broadcasting, reproduction on microfilms or in any other way, and storage in data banks. Duplication of this publication or parts thereof is permitted only under the provisions of the International Copyright Law and permission for use must always be obtained from Academy & Industry Research Collaboration Center. Violations are liable to prosecution under the International Copyright Law.

Typesetting: Camera-ready by author, data conversion by NnN Net Solutions Private Ltd., Chennai, India

Preface

The 6th International Conference on Computer Science and Information Technology (CoSIT 2019), was held in Dubai, UAE during February 23-24, 2019. The 6th International Conference on Artificial Intelligence and Applications (AIAPP 2019), 5th International Conference on Data Mining and Applications (DMA 2019), 5th International Conference on Software Engineering (SEC 2019) was collocated with 6th International Conference on Computer Science and Information Technology (CoSIT 2019). The conferences attracted many local and international delegates, presenting a balanced mixture of intellect from the East and from the West.

The goal of this conference series is to bring together researchers and practitioners from academia and industry to focus on understanding computer science and information technology and to establish new collaborations in these areas. Authors are invited to contribute to the conference by submitting articles that illustrate research results, projects, survey work and industrial experiences describing significant advances in all areas of computer science and information technology.

The CoSIT-2019, AIAPP-2019, DMA-2019, SEC-2019 Committees rigorously invited submissions for many months from researchers, scientists, engineers, students and practitioners related to the relevant themes and tracks of the workshop. This effort guaranteed submissions from an unparalleled number of internationally recognized top-level researchers. All the submissions underwent a strenuous peer review process which comprised expert reviewers. These reviewers were selected from a talented pool of Technical Committee members and external reviewers on the basis of their expertise. The papers were then reviewed based on their contributions, technical content, originality and clarity. The entire process, which includes the submission, review and acceptance processes, was done electronically. All these efforts undertaken by the Organizing and Technical Committees led to an exciting, rich and a high quality technical conference program, which featured high-impact presentations for all attendees to enjoy, appreciate and expand their expertise in the latest developments in computer network and communications research.

In closing, CoSIT-2019, AIAPP-2019, DMA-2019, SEC-2019 brought together researchers, scientists, engineers, students and practitioners to exchange and share their experiences, new ideas and research results in all aspects of the main workshop themes and tracks, and to discuss the practical challenges encountered and the solutions adopted. The book is organized as a collection of papers from the CoSIT-2019, AIAPP-2019, DMA-2019, SEC-2019

We would like to thank the General and Program Chairs, organization staff, the members of the Technical Program Committees and external reviewers for their excellent and tireless work. We sincerely wish that all attendees benefited scientifically from the conference and wish them every success in their research. It is the humble wish of the conference organizers that the professional dialogue among the researchers, scientists, engineers, students and educators continues beyond the event and that the friendships and collaborations forged will linger and prosper for many years to come.

Dhinaharan Nagamalai
David C. Wyld

Organization

General Chair

Dhinaharan Nagamalai
David C. Wyld

Wireilla Net Solutions, Australia
Southeastern Louisiana University, USA

Program Committee Members

Adnan Ahmad Mohammad
Ahmad Fadel Klaib,
Anas M.R. AlSobeh,
Asimi Ahmed,
Azizollah Babakhani,
Barbara Pekala,
Djemili Tolba Fatiha,
Federico Tramarin,
Felix Yang Lou,
Guilong Liu,
Hamid Ali Abed AL-Asadi,
Henrique Joao Lopes Domingos,
Isaac Agudo,
Jafar A. Alzubi,
Jose Luis Verdegay,
Jui-Pin Yang,
Mohammed Elbes,
Pietro Ducange,
Rafat Alshorman,
Samer Odeh Hanna,
Solomiia Fedushko,
Stefania Tomasiello,
Wladyslaw Homenda,
Yaseein Soubhi Hussein,

Yuan Tian,
Alessio Ishizaka,
Ali Asghar Rahmani Hosseinabadi,
Anazida Zainal,
Aysegul UCAR,
Babak Daghighi,
Bilal Hisham Ghanem,
Dhanamma Jagli,
Fatiha BOUSBAHI,
Hamid Ali Abed AL-Asadi,
Karim MANSOUR,
Malrey Lee,

Yarmouk University, Jordan
Yarmouk University, Jordan
Yarmouk University, Jordan
Ibn Zohr University, Morocco
Babo Noshirvani University of Tecnology, Iran
University of Rzeszow, Poland
University Badji Mokhtar Annaba, Algeria
University of Padova, Italy
City University of Hong Kong, China
Beijing Language and Culture University, China
Basra University, Iraq
New University of Lisbon, Portugal
University of Malaga, Spain
Al-Balqa Applied University, Jordan
University of Granada, Spain
Shih-Shien University, Taiwan
Al-Zaytoonah University, Jordan
eCampus University, Italy
Yarmouk University, Jordan
Philadelphia University, Jordan
Lviv Polytechnic National University, Lviv, Ukraine
University of Salerno, Italy
Warsaw University of Technology, Poland
Asia Pacific University of Technology &
Innovation, Malaysia
King Saud University, Saudi Arabia
University of Portsmouth, United Kingdom
Islamic Azad University Amol, Iran
Universiti Teknologi Malaysia, Malaysia
Firat University, Turkey
Azad University (West Tehran branch), Iran
Technical university of valencia, spain
V.E.S. Institute of Technology, India
King Saud University, Saudi Arabia
Basra University, Iraq
University Constantine, Algeria
Chonbuk National University, Republic of Korea

Mohamed Sathik,
Mohammad Mahmoud Abu Omar,
Nadine Akkari,
Omair Shafiq,
Ragab El Sehiemy,
Reza Ezzati,
Sabina Rossi,
Samer Al Martini,
Samy S. Abu Naser,
Sanjay Sharma,
Seyyed Reza Khaze,
Shifei Ding,
Smain FEMMAM,
Solomiia Fedushko,
Sultan Feisso,

Uduak Umoh,
Wided Oueslati,
Yuriy Syerov,

Sadakathullah Appa college, India
Al-Quds Open University, Palestine
Long Abdulatif University, KSA
Carleton University, Canada
Kafrelsheikh University, Egypt
Islamic Azad University, Karaj Branch, Iran
Ca' Foscari University of Venice, Italy
Abu Dhabi University, UAE
Al-Azhar University, Palestine
University Of London, Uk
Islamic Azad University, Iran
China University of Mining and Technology, China
UHA University, France
Lviv Polytechnic National University, Ukraine
Addis Ababa Science and Technology University,
Ethiopia
University of Uyo, Nigeria
École supérieure de commerce de Tunis, Tunisia
Lviv Polytechnic National University, Ukraine

Technically Sponsored by

Computer Science & Information Technology Community (CSITC)



Artificial Intelligence Community (AIC)



Soft Computing Community (SCC)



Organized By



Academy & Industry Research Collaboration Center (AIRCC)

TABLE OF CONTENTS

6th International Conference on Computer Science and Information Technology (CoSIT 2019)

Effectiveness Of U-Net In Denoising RGB Images01 - 10
Rina Komatsu and Tad Gonsalves

The Effect of Visualizing Role of Variable in Object Oriented Programming Understanding 11 - 20
Mabroukah Amarif and Sakeenah Ahmed

An Analyzing Algorithm Based On Learning And Searching In Chinese Medical Big Data21 - 32
*LUO Jie, ZHOU Ziyang, SONG Qiaolin and QIU Qin'nan *LI Zhimin*

The Implicit Path Cost Optimization in Dijkstra Algorithm Using Hash Map Data Structure 33 - 44
Mabroukah Amarif and Ibtusam Alashoury

In-Vehicle Camera Images Prediction By Generative Adversarial Network45-55
Junta Watanabe and Tad Gonsalves

6th International Conference on Artificial Intelligence and Applications (AIAPP 2019)

Online Knowledge-Based Expert System (Kbes) For Psychological Diseases Diagnosis 57-71
Ahmad A. Al-Hajji, Fatimah M. AlSuhaibani and Nouf S. AlHarbi

Mixtures Of Regression Curve Models For Arabic Character Recognition73-82
Abdullah A. Al-Shaher

5th International Conference on Data Mining and Applications (DMA 2019)

Detection of Hate Speech In Social Networks: A Survey On Multilingual Corpus83 - 100
Areej Al-Hassan and Hmood Al-Dossari

5th International Conference on Software Engineering (SEC 2019)

Paralling Verification Execution With Verifying Algebra In A Cloud Environment101 - 113
Kan Luo, Siyuan Wang, An Wei, Wei Yu, Kai Hu

EFFECTIVENESS OF U-NET IN DENOISING RGB IMAGES

Rina Komatsu and Tad Gonsalves

Department of Information & Communication Sciences, Faculty of Science &
Technology, Sophia University, Tokyo, Japan

ABSTRACT

Digital images often contain “noise” which takes away their clarity and sharpness. Most of the existing denoising algorithms do not offer the best solution because there are difficulties such as removing strong noise while leaving the features and other details of the image intact. Faced with the problem of denoising, we tried solving it with a Convolutional Neural Network architecture called the “U-Net”. This paper deals with the training of a U-Net to remove 3 different kinds of noise: Gaussian, Blockiness, and Camera shake. Our results indicate the effectiveness of U-Net in denoising images while leaving their features and other details intact

KEYWORDS

Deep Learning, Image Processing, Denoising, Convolutional Neural Network, U-Net.

1. INTRODUCTION

Today’s imaging devices such as digital cameras, smart phones and video cameras invade our lives, we become constantly aware of the need for more and more clear images. However, there is a rather formidable obstacle named “noise” which makes the digital images taken by these devices hard to appreciate. Noises are generated from transferring images in the digital media, due to human error while taking pictures, and while restoring compressed images back to their original size. The kinds of noises are various: some of them make the image gloomier, some make them sandy and so on. There are denoising solutions to deal with each kind of noise. However, the denoising process may take time in detecting the kind of noise and applying the relevant correcting algorithm with optimized parameters. There are other limitations, too. For example, if the noise is too strong, it is hard to recognize the objects in the image.

To remove various noises no matter how strong, technology like the human brain which can distinguish between noise and the original image is needed. In recent years, AI deep learning algorithms and techniques are rapidly progressing to handle recognition, segmentation, reproduction of images, etc. Using deep learning techniques, we investigated if it is possible to build a model that can effectively denoise noisy input images and get clear output images without any traces of noise.

In this paper, we picked up the CNN architecture “U-Net” as a typical denoising deep learning model and developed our own “Reformed U-Net” to handle strong noise in the images. In Section 2, we introduce 3 kinds of noises we used as denoising targets. Section 3 does a brief U-Net overview and introduces comparative studies of the 2 models based on U-Net; Section 4 and 5 describes programming environment and experimental setup; Section 6 indicates denoising results; Section 7 concludes this study.

2. OVERVIEW: NOISE USED IN OUR EXPERIMENT

In our study, 3 kinds of noises: Gaussian noise, Blockiness and Camera shake are selected as the target for image correction. This section introduces these types of noises and their effects.

2.1. Gaussian Noise

Gaussian Noise makes image quality sandy due to the problem in electronic circuits or sensors. As the value consists of Gaussian distribution covering the original image, RGB pixels in the original image are collapsed and lose the special information such as contrast, brightness and so on. Figure1 shows the original image and the one in which Gaussian noise is added.

In image proceeding field, filtering process is usually used to reduce the effect of this noise [1]. This process is convolute to noisy image with a filter such as median filter, gaussian filter etc. The problem with this solution is filtered image become less brightness since the pixels near the noise is also convoluted.



Figure1. Adding Gaussian noise

2.2. Blockiness

When uploading images on to internet servers, compression becomes necessary because of their large quantity of bytes. When restoring the compressed images, block effect comes into play blurring the smooth edges. Figure 2 shows generating blockiness by resizing the original image into a smaller size and then returning the compressed image back to the original size. Filtering process is also used to extract edge information [2].



Figure 2. Generating Blockiness

2.3. Camera Shake

Camera shake happens when we take pictures with trembling or non-steady hands. As a result, the output becomes a picture which is out of focus and it is hard to detect objects in such blurred

images (Figure 3). There is the technique named blind deconvolution which correct images without hardware like gyro sensor, only blurred image. As the example study in detecting blur kernel, output Fourier transformed image from blurred one and get blur vector and length [3] is picked up.



Figure 3. Generating Camera shake

3. OVERVIEW: DEEP LEARNING MODELS

In deep learning field, Vincent et al proposed learning architecture named Denoising Autoencoder [4]. This architecture train neural network model to be able to reconstruct clear data from noisy input. As a denoising neural network model, this section introduces 2 models.

3.1. U-Net

In our study, U-Net is adopted as the generator model. U-Net is the one of the Convolutional Neural Network (CNN) architectures proposed by Ronneberger et al. The architecture of the network shown in Figure 4 describes the alphabet “U”. The left side which consists of the Convolution layers is called the contracting path and the right side which consists of Deconvolution layers is called the expansive path. As explained in [5], the contracting path gets the context, while the expansive path holds the precise localization.

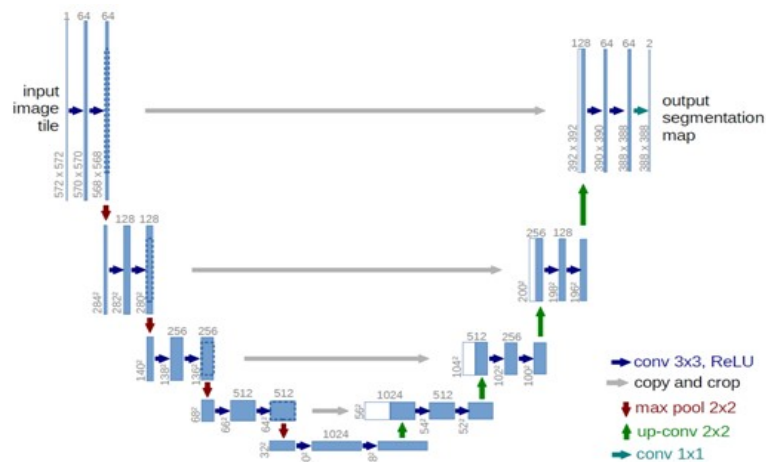


Figure 4. Architecture of U-Net (quote from Fig.1. from [5])

This U-Net is not only used in image segmentation. Isola and others who propose image architecture “px2px”, utilizes U-Net as generator model to transform daylight scene into night scene, to generating a map from an aerial photograph, and painting from an edge image and so on [6].

Referring to Figure 4, the sizes of the input and the output image are different. When convolute to input, output size relies on the parameter: filter size, stride size and padding size. Under formula (1) shows the output sizes of height (h_{output}) and width (w_{output}). (Formula (1) referred from a part of document [7])

$h_o = \frac{h_i + 2h_p - h_k}{s_y} + 1$ $w_o = \frac{w_i + 2w_p - w_k}{s_x} + 1$	(1)
<p>$(h_o, w_o) :=$ output image size $(h_i, w_i) :=$ input image size $(h_k, w_k) :=$ filter size $(s_y, s_x) :=$ stride direction $(h_p, w_p) :=$ padding size</p>	

To be able to compare images of identical size, we reconstruct U-Net by adjusting several parameters in each layer. Figure 5 shows the U-net which outputs same sized images used in our study.

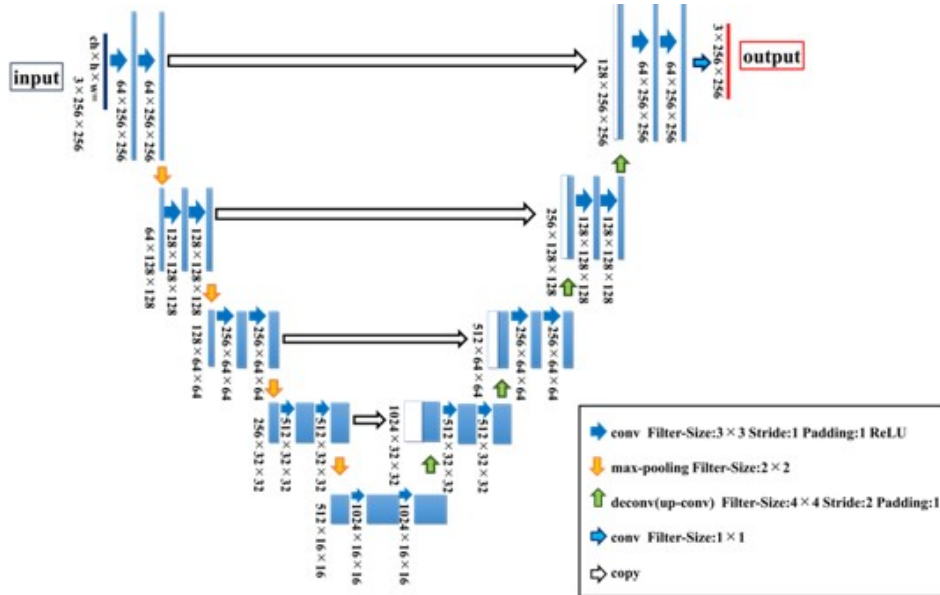


Figure 5. Architecture of U-Net used in this study

3.2. Reformed U-Net

According to some techniques adopted by DCGAN: Deep Conditional GANs [8], performance is enhanced by replacing pooling layers with fractional-stride convolutions and using Batch Normalization (Batch Normalization normalizes layer's input and helps higher learning rate [9].) to each layer's output in model. Following these techniques, we reformed the U-Net shown in Figure 5. We named this proposed U-net "Reformed U-Net". Figure 6 shows the architecture of Reformed U-Net.

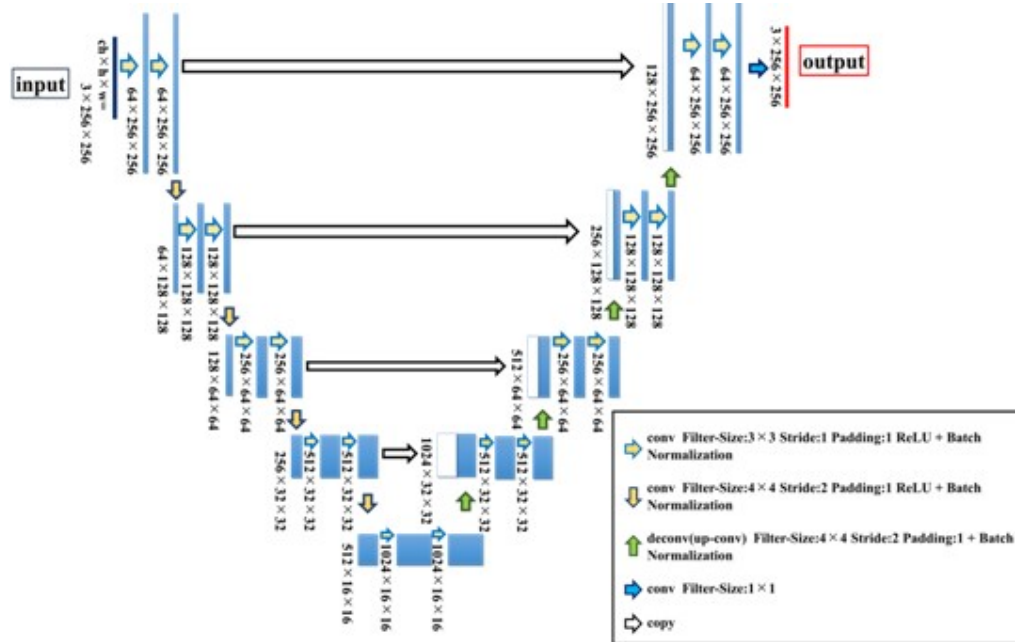


Figure 6. Architecture of Reformed U-Net

4. EXPERIMENT ENVIRONMENT

4.1. Image Dataset

As the materials to adding noise, we choose the ADE20K dataset [10]. This dataset was used in the image segmentation competition and is rich in the variety of landscape pictures. This dataset has been split in 20,210 images for training and 3,352 images for testing in advance.

4.2. Programming

Our study tried using “chainer” [11] as the deep learning framework. This framework is used from building deep learning models to training and testing. Also, to deal with image processing in programming, the image library “OpenCV” is employed. Noisy images are generated using OpenCV, while inputting the model and training it to be able to output clear images is performed using the functions provided by the chainer framework.

5. EXPERIMENT

5.1. Creating Noisy Images

First, all images, training as well as testing were resized to 256×256 pixels. Noise is then added to each resized image. Gaussian noise is generated by overlapping a clear image with Gaussian distribution whose parameter σ is in the range [15, 50]. In generating Blockiness, images are first resized in the scale [1/4, 1/2] and then reframed back to the original 256×256 px size. Camera shake is generated on purpose by overlapping a few copies of the same images, each with a varying focus.

5.2. Training

Next is training each model to generate clear images from noisy ones. Each of the 3 steps described below counts as 1 epoch. Training is carried on for 100 epochs with mini batches of 15 images per batch.

Step 1. Load noisy image as the input of the model and non-noisy clear image as the target.

Step 2. Normalize noisy image by dividing by 255. Then, input to the model and get output.

Step 3. Calculate loss by Mean Absolute Error (MAE) between output and target and update the parameters in U-Net by optimizer Adam [12] which is attached chainer's library.

5.3. Evaluating The Trained Model

To evaluate how the trained model could render clear images from the input noisy images, we employed 2 different estimate methods. The 1st method relies on the loss changes which calculates the MAE during the training to estimate whether the output from the model comes near to the target. The 2nd method is visualizing the output in RGB image and comparing it with the target image by calculating the MAE from pixel values between the visualized output image and the target image. Here we use PNSR: Peak signal-to-noise ratio which is the criterion for determining how close the output resembles the target.

6. RESULTS

6.1. Loss Changes

To evaluate whether there was an over-fitting in the trained, we compared the loss changes using the training and the testing datasets. Figure 7 shows the graph of loss change recorded from epoch 1 to 100. From the start to finish, both the models steadily reduce the value of the loss function.

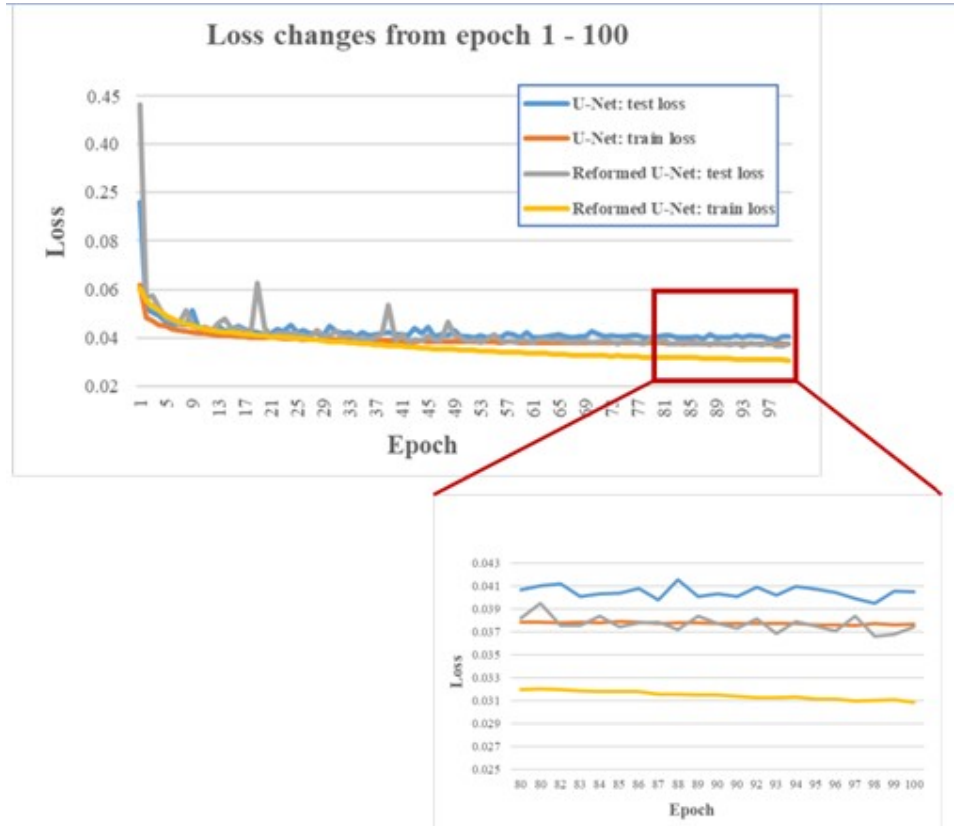


Figure 7. Loss changes from epochs 1-100 in training and testing

Referring to the loss in training and testing in Figure 7, we notice that each loss becomes less and less as the epoch proceeds. Both the models did not show any overfitting, a phenomenon in which the loss change in the case of testing remains relatively flat while that of the training phase steadily decreases. On finishing training and testing the model, the loss of U-Net settled on 0.038 in training and 0.041 in testing, while the Reformed U-Net settled on 0.031 in training and 0.037 in testing.

6.2. Visualizing Output Of The RGB Image

Since both the models could decrease the loss in training and testing, we should evaluate how clearly the model succeeds in removing noise from a given noisy image. Figure 8 shows the visualized output result using the model trained through 10, 50, and 100 epochs. Using the PNSR and MAE criteria, both models could denoise image as PNSR increases and MAE decreases compared to the noisy image. Also, the model trained over a greater number of epochs produced improved results than the one with a smaller number of epochs.

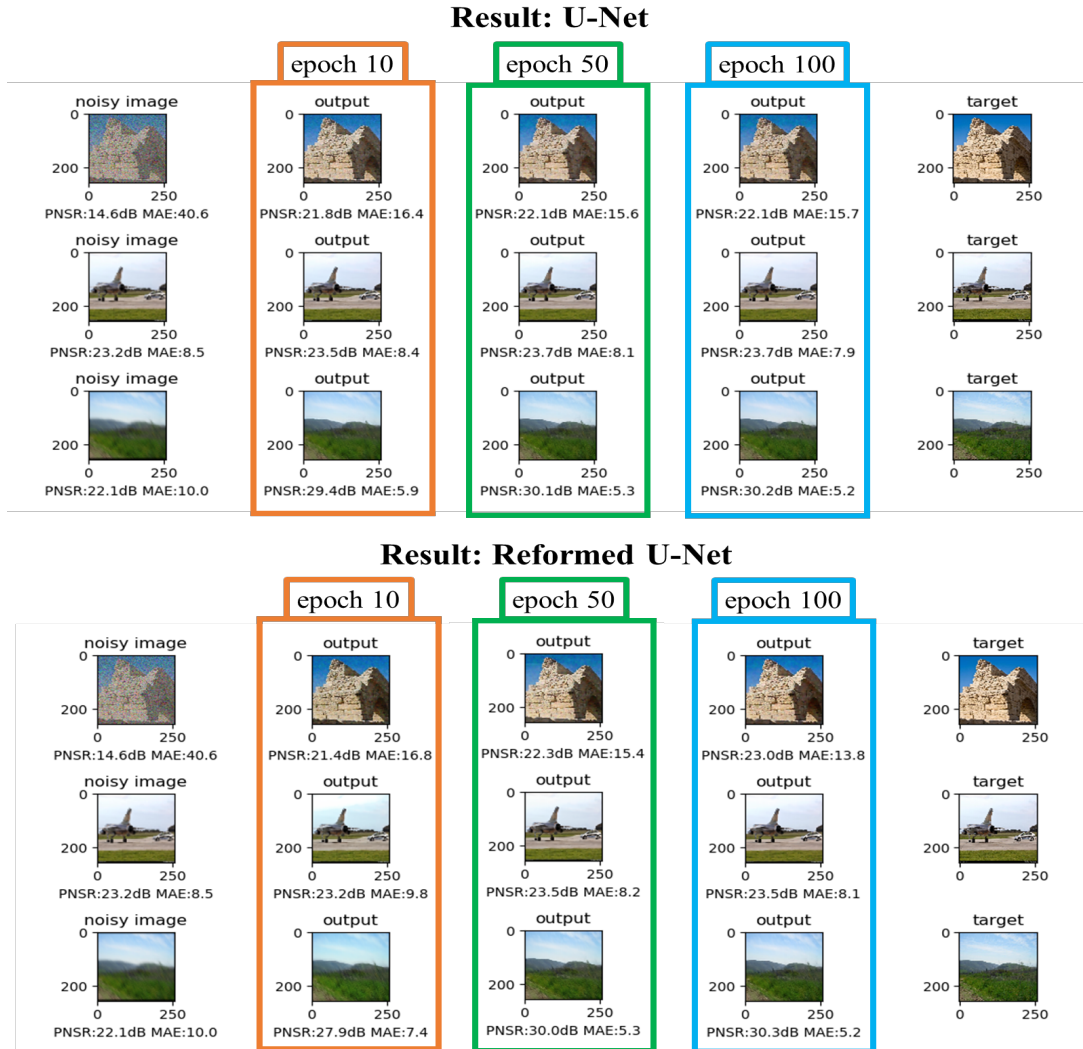


Figure 8. Visualized output result

6.3. Comparison In Dealing With Strong Noise

The results in section 6.1 and 6.2 indicate that the deep learning model based on U-Net has enough ability to get rid of different kinds of noise. This study picked up 2 models, the U-Net and the Reformed U-Net. Which model is better for denoising strong noise? To ascertain this, we had additional experiment steps described below:

Step1. Add strong noise to test images by adjusting the parameters described in section 5.1.

Step2. Input strong noisy image which has been normalized and get the output.

Step3. Calculate loss by MAE.

We try denoising each kind of noise one by one. Table 1 shows the result of loss. We observe that the Reformed U-Net is better than U-Net in dealing with strong noise.

Table 1. Comparison result dealing with strong noise.

	Gaussian Noise	Blockiness	Camera Shake
U-Net	0.0676	0.0423	0.0456
Reformed U-Net	0.0605	0.0422	0.0426

7. CONCLUSION

Our results show that Deep learning networks like the U-Net are effective in denoising RGB images. Further, the Reformed U-Net we proposed in this study can deal with strong noise more effectively than the U-Net. The reason why the neural network model based on U-Net succeeded in denoising might be associated with the connections between contracting path and expansive path. According to Mao Xiao-Jiao and others who proposed the model architecture “skip connections” (skip connection is linking between corresponding convolution and deconvolutions), skip connection has effect to gradient vanishing problem and be able to pass image details from convolution layers to deconvolutions layers which roles recovering noisy input [13]. Like U-Net which has linking architecture could learn denoising in stable and get clear output.

However, the effect in denoising blockiness was not more outstanding than the other noises. It seems to be associated with the measure used to obtain loss. The Mean Absolute Error calculates the loss depending only on the value in each pixel and not the visual aspect. To consider the visual aspect, we may need the discriminator model which learns through training to distinguish the fake images (i.e. output from the generator model) and the real (i.e. clear image before adding noise).

REFERENCES

- [1] Masanao Koeda, Takayuki Nakamura & Etsuko Ueda, (2014) “Introduction to image Processing with OpenCV, 2nd ed”, Kodansya, (Japanese).
- [2] Y. L. Lee, H. C. Kim & H. W. Park, (1998) “Blocking effect reduction of JPEG images by signal adaptive filtering”, IEEE TRANSACTIONS ON IMAGE PROCESSING, Vol.7, No.2, pp.229-234.
- [3] Kenichi Yoneji, Masayuki Tanaka & Masatoshi Okutomi, (2005) “PSF Parameter Estimation for Restoration of Linear Motion Blurred Image”, Computer Vision and Image Media (CVIM), vol.2005, no.38, pp47-52.
- [4] Pascal Vincent, Hugo Larochelle, Yoshua Bengio & Pierre-Antoine Manzagol, (2008) “Extracting and composing robust features with denoising autoencoders”, Proceedings of the 25th international conference on Machine learning, ACM, pp1096-1103.
- [5] Olaf Ronneberger, Philipp Fischer & Thomas Brox, (2015) “U-net: Convolutional Networks for Biomedical Image Segmentation”, International Conference on Medical image computing and computer-assisted intervention (MICCAI), Springer, Vol. 9351, pp234-241.
- [6] Phillip Isola, Jun-Yan Zhu, Tinghui Zhou & Alexei A. Efros, (2017) “Image-to-Image with Conditional Adversarial Networks”, 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp5967-5976.
- [7] `chainer.functions.convolution_2d`, URL: http://docs.chainer.org/en/stable/reference/generated/chainer.functions.convolution_2d.html#chainer.functions.convolution_2d, (access data: 23/01/2019)

- [8] Alec Radford, Luke Metz & Soumith Chintala, (2015) “Unsupervised representation learning with deep convolutional generative adversarial networks”, arXiv preprint arXiv:1511.06434.
- [9] Sergey Ioffe & Christian Szegedy, (2015) “Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift”, arXiv preprint arXiv:1502.03167.
- [10] ADE20K dataset, URL: <http://groups.csail.mit.edu/vision/datasets/ADE20K/>, (access date: 17/12/2018).
- [11] Seiya Tokui, Kenta Oono, Shohei Hido & Justin Clayton, (2015) “Chainer: a next-generation open source framework for deep learning”, Proceedings of workshop on machine learning systems (LearningSys) in the twenty-ninth annual conference on neural information processing systems (NIPS), Vol.5, pp1-6.
- [12] Diederik P. Kingma & Jimmy Lei Ba, (2014), “Adam: A Method for Stochastic Optimization”, arXiv preprint arXiv:1412.6980.
- [13] Mao Xiao-Jiao, Chunhua Shen & Yu-Bin Yang, (2016), “Image Restoration Using Very Deep Convolutional Encoder-Decoder Networks with Symmetric Skip Connections”, Advances in neural information processing systems, pp2802-2810.

THE EFFECT OF VISUALIZING ROLE OF VARIABLE IN OBJECT ORIENTED PROGRAMMING UNDERSTANDING

Mabroukah Amarif and Sakeenah Ahmed

Department of Computer Sciences, Sebha University, Sebha, Libya

ABSTRACT

The role of any variable is interpreted as the required task or performance of it in any part of a program. This role contributes to the easy understanding of the program and thus formulates it clearly and unambiguously. Many novice programmers face various difficulties in understanding programming, especially Object Oriented Programming. This research adopts the design of a visualization tool which includes visual model that shows the role of the reference variable (an object) within a Java program to enhance comprehension understanding for novice programmers. The model enables them to interact and thus formulate an object-oriented program in an intuitive and clear way. Based on the actual experimentation, the effectiveness of this model is improved and the importance of this research in the field of object programming is demonstrated.

KEYWORDS

Role of variable, object oriented programming, visualization, understanding

1. INTRODUCTION

The concept of the role of variable is a modern in view with respect of the lack of sufficient known about it. In the last decade, it has been used to facilitate and improve programming education [1, 2]. It is an educational implement distinct style, if used properly, to clarify the structure and meaning of the program [3]. The roles of variables contribute to the outputs of the program, indicating the achievement of the program goal or the whole system [4]. Sajaniemi et al. stated that the role of the variable certainly provide novice programmers with a new conceptual framework that enables them to rationalize and mental processing their programs in a way similar to understanding of a good code, they also stated that the role of any object-oriented variable can be defined by its properties and behavior during the program [5, 7].

Regarding of the view of experienced programmers, it was observed that their knowledge is implicitly embodied in the cognitive structures or diagrams structures for any program [2], while the new or novice programmers often find it difficult to understand a program question because they are unable to identify a variable or more, or sometimes they couldn't understand the difference between one variable and another [3, 6]. The reason lies in their inability to visualize the desired role of the variable that is determined in the program.

The variable has a dynamic property. In other words, it contains a value and then contains another value, thus successive values obtained (variable behavior). Therefore, it is called a variable that is characterized by a change in its value in any part of the program as required.

Looking at the object oriented programming languages; many novice programmers face many difficulties, especially in the process of correctly formulating the program [4, 7, 8]. It is realized that the most difficult part lies in the ambiguities of the variables analysis.

The data structures description of procedural programming are assigned to the programmer in terms of their inclusion, logical compilation, data types and functions, while in the object-oriented programming languages, a lot of language structures are part of the same language, which is ready and suitable for the formulation of large programs and delivered by a team of programmers. In fact, object oriented programming is an influential and effective model at present time. It's special because they have a range of tools and techniques that enable software engineers to build reusable software systems that meet user needs. The most important of these tools are class and object which used as a storage of data and operations on them through the exchange of messages between the program and objects and between objects themselves rather than functions call which found in procedural programming languages. In the other side, these tools may overwhelm the initial programmer because of their large numbers, or the difficulty of dealing with them in principle.

In this paper, we describe our developed tool called LearnOOP for teaching/learning OOP Java programming language. The tool development adopts the role of variable concept and visual forms to reflect the behaviour of variable and its roles. We also describe our experience as an experimental procedure and evaluate the developed tool using empirical evaluation approach. The following section describes the most related works to our paper while section 3 explains the developed tool description. Experimental procedure and results are described in section 4. A discussion of this paper is explained in section 5 and the conclusion is provided in section 6.

2. RELATED WORK

According to our currently search and based on what we found, a few tools have been developed for teaching/learning programming languages. Some of them adopt role of variable concept such as PlanAni [9] and ViLLE [10,11], others are not, such as OOP-ANIM [12] and Jeliot [13,14,15]. PlanAni and ViLLE tools are used for procedural programming understanding while OOP-ANIM and Jeliot are used for object oriented programming comprehension. Moreover, the animation and visualization techniques have been adopted in all of these tools but in different ways. Only PlanAni tool visualizes the role of variable from a real world component, for example; counter variable is visualized as a step foot.

ViLLE tool gives a description of role of variable in text-based form. It explains the changing values of variable at each stage of program executing.

Although the OOP-ANIM and Jeliot tools explain the object oriented programming concepts, none of them include the concept of role of variables. OOP-ANIM uses UML diagrams as a visual form of program parts, in the other side; Jeliot tool uses boxes and text form for program executing visualization.

Different studies have been carried out for evaluating these tools to know how much does role of variable concept effect in learning and understanding. From the literature, Sajaniemi et al. have conducted various studies [1, 3, 5, 7, 9]. Their results showed a positive effect of roles of variables in understanding of programming languages. They also agree that the concept of role must be enhanced by visual description that reflects the reality of its behavior.

Another research has been carried out by Shi et al. [6]. They considered the visualization and animation effects of role of variable in building a novice program. They carried out an experiment on 55 students of computer sciences. They divided the students into two groups; the first learning group learned programming in the traditional way based on the role. The second group learned with the support of PlanAni tool based on the visualizing roles of variables. Their data analysis showed a general improvement of the level of SOLO and high approval for the second group.

Al-Barakati et al. have conducted an experimental study [17]. They used a PlanAni tool to teach students an introductory programming course. Their results showed in scores on a post-test used to evaluate the effect of visual role of variable in understanding with 91 students divided into three groups. The results showed that the visual role of variable groups significantly outperformed the non visual group.

A case study have been carried out by Rajala et al. [10] on ViLLE tool focusing on the effectiveness of visualization of role of variable on learning basic programming concepts. They were randomly divided students into two groups. The control group used traditional textual material during the course, while the treatment group used the same material extensively with interactive examples using the ViLLE tool. The study found that the ViLLE tool improves student learning without prior programming experience. They also found that ViLLE tool benefit novice learners more than learners who have some previous experiences.

Various studies have been carried out using Jeliot tool during the development of the tool versions [15, 16]. Regarding of learning with visualization and animation within the program execution, the results show positive effect. They also concluded that the educational tool must be incorporated into classes and assignments rather than once used as an aid in teaching.

Although there are tools in object programming for increasing students' understanding, they lack of the concept of variables roles and if variable roles are included, they still need to be visualize for enhancing students' understanding. All of these motivate us to propose and develop our own tool which adopts the visualization of variables roles using real world objects for more students' perception and understanding of OOP java language.

The main goal of this paper is to justify the positive effect of visualizing the role of variable in OOP Java programming understanding through our developed tool called LearnOOP and evaluate quantitatively the effectiveness of it. For the purpose of this paper, effectiveness refers to the ability of this tool in enhancing student's perception and understanding. This goal is evaluated using an empirical evaluation approach (traditional teaching without a tool vs. using LearnOOP tool). The question is, "*Is teaching using LearnOOP tool which includes the visualization and animation of role of variables more effective than traditional teaching medium?*".

3. LEARNOOP TOOL DESCRIPTION

LearnOOP is a tool used for teaching/learning Java OOP programming language. It contains the description of object and how being created from a special class. It adopts the role of variable concept through an object description and different operations on it. The description of object is given by visualization and animation of his attributes and behaviour. The tool includes various examples of objects from the real world to increase the ability of understanding OOP. For an example; an object car1 is created from a class Car. Figure 1 shows the interface of the tool and the object car1 example.

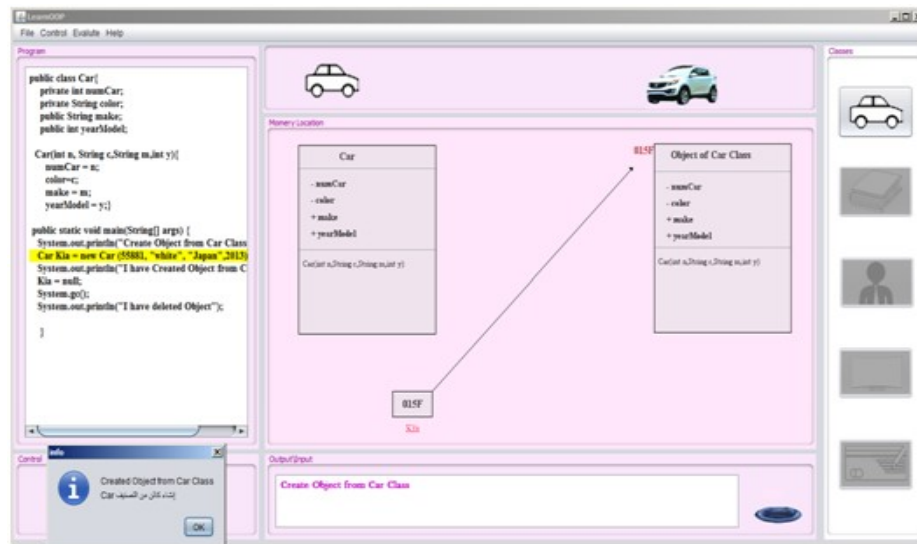


Figure 1. LearnOOPTool user interface

4. LEARNOOP TOOL EVALUATION

We carried out an experiment consists of 40 students of the same lesson taught to the undergraduate Computer Science in Java Programming I (OOP) course at Sebha University of Libya during the semester I of 2017-2018 year. The students never studied OOP before. The experiment was conducted in two stages where each stage uses a different learning medium approach; the first stage uses only text-based materials (no visualization and animation of roles of variables), in the second stage, we divide the students into two groups randomly. Each group consists of 20 students. The first group called the control group which continuous in traditional teaching way while the other uses LeanOOP tool (visualizing role of variable concept). The students will be given a same test throughout the two stages. They are allowed to improve their answer after each stage. The results of the tests after each stage of the medium approach are compared.

The same topics of the lessons are given during all of the two stages. These topics are: class and object concepts, operations on object such as create, delete, copy, display, and modify object attributes. It also covers the inheritance lesson.

In this experiment, the tool SPSS [18] is used to statistically evaluate the effectiveness of LearnOOP tool using t-test and p-value.

4.1. Experimental Procedure

A total of 40 students participated in the experiment. The students are second year of Computer Science students (undergraduate students) at Sebha University of Libya. We follow the pre-test to post-test accuracy [19, 20] in order to evaluate the effectiveness of LearnOOP tool. At the first stage, all of the students were given the same lesson using traditional teaching medium followed by a pre-test. At the second stage, the students were divided into two equal groups randomly; the first group continuous with traditional teaching medium and the second group were introduced to LearnOOP tool followed by a post-test for both groups.

The experiment was controlled by delivering the same lessons to all of the students by the same teacher during one semester long. The semester was divided into two equal stages of time. In the first stage, only text-based materials were used during the lesson time with the help of electronic slides. At the end of the stage, the students were given a pre-test of three OOP questions with a time limit of one hour to answer them.

In the second stage, after the pre-test, students were divided into two equal groups; group A and group B. group A continued with traditional teaching medium. group B introduced to LearnOOP tool and to its visual interface. They were asked to experiment with some examples of real world objects such as Car, Employ, Television, Book and Account. They were also asked to experiment with the concepts of class and object, operations on objects and inheritance. At the end of the session, both groups were given a post-test of different OOP questions regarding of the same objects with a time limit of one hour to answer them.

To control the tasks performance, the students were not allowed to consult books or use any materials. Then the results of pre-test and post-tests were compared.

4.2. Experimental Results

To determine the effectiveness of LearnOOP tool, a pre-test and post-test accuracy is used. Table 1 describes the students' scores of group B for the pre-test before using LearnOOP tool and post-test after using LearnOOP tool. Notice that the maximum score for each student is 15.

Table 1. The students' scores (group B) of pre-test and post-test

No.	Pre-test scores Without a tool	Post-test scores Using LearnOOP tool
1	2	10
2	2	11
3	5	12
4	5	12
5	2	11
6	2	10
7	11	14
8	2	9
9	1	4
10	3	7
11	2	10
12	2	9
13	4	12
14	1	7
15	2	10
16	2	10
17	1	4
18	1	4
19	3	7
20	1	4
total	54	177

The means (averages) of students' scores of pre-test and post-test are calculated. Table 2 describes the means values of the group tested.

Table 2. The students' scores means of pre-test and post-test (group B)

Time	Treatment	No.	Mean
Sebha University	Without a tool	20	2.7
	Using LearnOOP tool	20	8.85

According to the chart analysis of the means values, it's clear that the mean value using LearnOOP tool is higher than the mean value without using the tool. Figure 2 explains the idea.

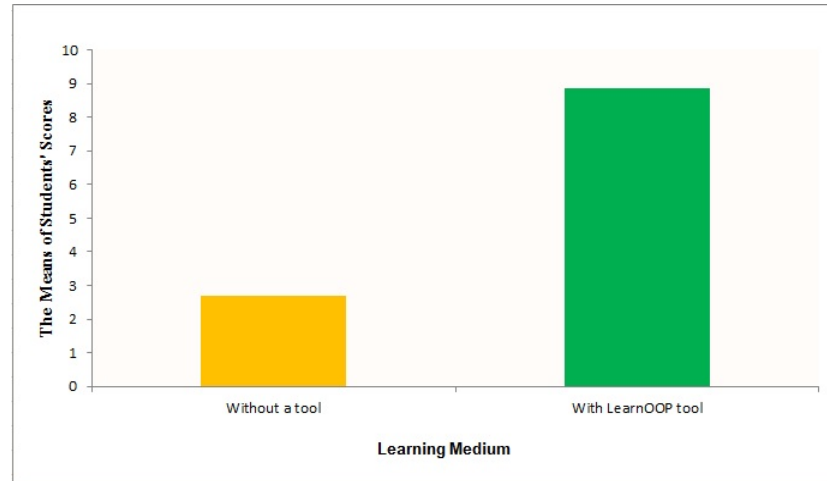


Figure 2. The means of the students' scores (group B)

The adopted statistical analysis of this experiment is that:

- Null Hypothesis (H_0): the conducted hypothesis is that there is no difference in the mean of pre-test and post-tests scores. In other words, the pre-test and post-tests scores will have equal means.
- Alternative hypothesis (H_1): the alternative hypothesis is that there is at least one difference in the mean of the pre-test and post-test scores in the group tested.
- p-value: the return value of the statistical test which indicates the probability of getting a mean difference between the pre-test and post-test as high as what is observed by chance. The lower the P-value, the more significant difference between the groups. The typical significance level that has been chosen in this experiment is 0.001.
- t-test: this test was run on the pre-test and post-test scores. In this experiment, the result t-test shows that there is a difference between the pre-test and post-test according to the p-value which is 0.000 and less than the significance level 0.001. Table 3 shows the result of t-test.

Table 3. The results of t-test

Treatment	No. of student	Mean	p-value	t- test
Without a tool	20	2.7	0.000	LearnOOP tool > No tool
Using LearnOOP Tool	20	8.85		

The test shows that there is a difference between using the LearnOOP tool and without using it based on the p-value which equal to 0.000. The p-value is less than the significance level (0.001) and that means the improvement from pre-test to post-test is statistically high significant.

Table 4 describes the students' scores of group A (without a tool) for the post. Notice that the maximum score for each student is 15.

Table 4. The students' scores (group A) of post-test

No.	Post-test scores without a tool
1	5
2	2
3	10
4	3
5	3
6	2
7	7
8	2
9	6
10	7
11	4
12	4
13	4
14	4
15	2
16	3
17	2
18	1
19	10
20	13
total	94

Again, The students' scores means of post-test for group A is calculated. Table 5 describes the means of both groups tested.

Table 5. The students' scores means of post-test for group A and B

Time	Treatment	No.	Mean
Sebha University	Without a tool (group A)	20	4.7
	Using LearnOOP tool (group B)	20	8.85

According to the chart analysis of the means values, it's clear that the mean value of group B, which uses LearnOOP tool, is more higher than the mean value of the group A, which does not use it. Figure 3 explains the idea.

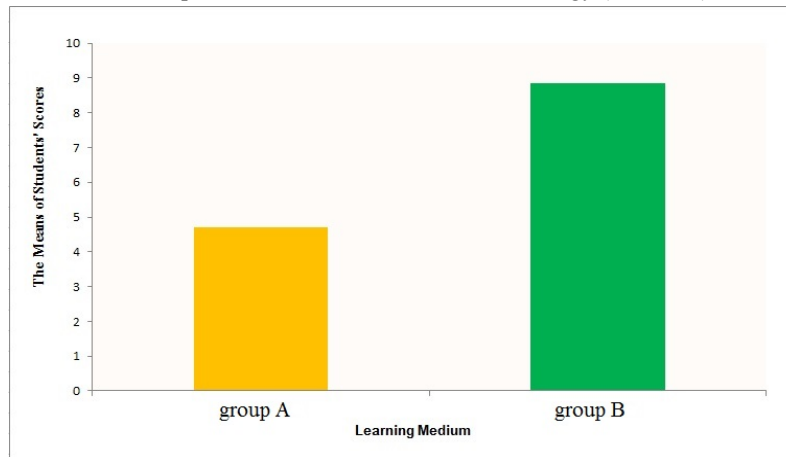


Figure 3. The means of the students' scores between group A and B

The same previous statistical analysis is adopted for calculating the t-test; in this time, the test was run on the post-test of the two groups; A and B scores. The result t-test shows that there is a difference between the post-test of group A and post-test of group B according to the p-value which is 0.000 and less than the significance level 0.001. Table 6 shows the result of t-test.

Table 6. The results of t-test

Treatment	No. of student	Mean	p-value	t- test
Without a tool (group A)	20	4.7	0.000	LearnOOP tool > No tool
Using LearnOOPTool (group B)	20	8.85		

5. DISCUSSION

The results in this experiment indicate that LearnOOP Tool is more effective and efficient than traditional learning medium. According to our hypothesis testing, there is a significant difference between using LearnOOP Tool as a teaching/learning medium and text-only material. It shows that learning with visualizing role of variable significantly outperformed text-only material. The students realize many facts about object programming, the most important of which is that the object name indicates a variable that contains the object data address in the memory. They also learn from the tool how to extract the object's attributes through visual form. Meanwhile, the overall improvement of enhancing the students' ability for understanding the OOP Java using LearnOOP Tool is demonstrated and achieved. Compared with the previous studies, we almost agree with them on the need for the use of supporting education tools. We also agree on the need of adopting the role of variable concept for clarifying the programming and coding of different programming languages.

6. CONCLUSION AND FUTURE WORK

Regardless of the advancement in the area of educational techniques, it needs to be further tested with more experimental assessment and empirical evaluation, especially of using the teaching/learning visualization and animation tools during lectures and classrooms. Currently, a few researches dealt with the problem of the lack of using these kinds of tools. Furthermore, studies have shown that the role of variable concept educationally enhanced students'

understanding if they were supported by visualizing forms from the real world. This paper was motivated by these observations. In particular, this paper suggested more experiments of other tools through empirical evaluation in order to improve their effectiveness and teaching/learning support in the field of programming understanding. Our future work will consider adding more OOP features to LearnOOP tool such as polymorphism. Showing the relationship between entity and objects is also an interested area of OOP programming languages. Another future work is running more experiments and empirical evaluation for LearnOOP tool and other related tools.

REFERENCES

- [1] Sajaniemi, J., "Roles of variables and learning to program," in Proc. 3rd Panhellenic Conf. Didactics of Informatics, Jimoyiannis A (ed) University of Peloponnese, Korinthos, Greece, 2005.
- [2] Sorva, J., Karavirta, V. & Korhonen, A., (2007) "Roles of variables in teaching," Journal of Information Technology Education, vol. 6, pp. 407-423.
- [3] Byckling, P, Gerdt, P., & Sajaniemi, J., "Visualizing roles of variables to novice programmers," in Proceedings of the 14th Annual Workshop of the PPIG'02, 2002, pp. 111-127.
- [4] Reenskaug, T., "Roles and Classes in Object Oriented Programming", 2007.
- [5] Kuittinen, M., & Sajaniemi, J., "Teaching roles of variables in elementary programming courses," ACM SIGCSE Bulletin, vol. 36, pp. 57-61, 2004.
- [6] Shi, N., Min, Z., & Zhang, P., (2017) "Effects of visualizing roles of variables with animation and IDE in novice program construction," Telematics and Informatics.
- [7] Gerdt, P., & Sajaniemi, J., "Roles of Variables in Object Oriented Programming", OOPSLA'05, San Diego, California, USA, ACM 1595931937/05/0010, October 16–20, 2005.
- [8] Xinogalos, S., Sartatzemi, M., & Dagdilelis, V., "Studying Students' Difficulties in an OOP Course Based on BLUEJ", Proceedings of the 9th IASTED International Conference, Computers and Advanced Technology in Education, Lima, Peru 2006.
- [9] Sajaniemi, J., & Kuittinen, M., "Program Animation Based on the roles of Variables," in Proceedings of the 2003 ACM symposium on Software Visualization, 2003.
- [10] Rajala, T., Laakso, M.-J., Kaila, E., & Salakoski, T., (2008) "Effectiveness of Program Visualization: A Case Study with the ViLLE Tool," Journal of Information Technology Education, vol. 7.
- [11] Laakso, M.-J., Malmi, L., Korhonen, A., Rajala, T., Kaila, E., & Salakoski, T., (2008) "Using roles of variables to enhance novice's debugging work," Issues in Informing Science and Information Technology, vol. 5, pp. 281-295.
- [12] Esteves, M., & Mendes, A., "OOP-Anim, a system to support learning of basic object oriented programming concepts," in Proceedings of CompSysTech'2003-International Conference on Computer Systems and Technologies. Sofia, Bulgaria, 2003.
- [13] Levy, R., Ben-Ari, M., & Uronen, P. A., (2003) "The Jeliot 2000 program animation system," Computers & Education, vol. 40, pp. 1-15.
- [14] Ben-Ari, M., Myller, N., Sutinen, E., & Tarhio, J., (2002) "Perspectives on program animation with Jeliot," in Software Visualization, ed: Springer, pp. 31-45.
- [15] Moreno, A., Myller, N., Sutinen, E., & Ben-Ari, M., "Visualizing programs with Jeliot 3," in Proceedings of the working conference on Advanced visual interfaces, 2004, pp. 373-376.

- [16] Čisar, S., M., Radosav, D., Pinter, R., Čisar, P., (2011) "Effectiveness of Program Visualization in Learning Java: a Case Study with Jeliot 3" *Int. J. of Computers, Communications & Control*, Vol. VI, No. 4 (December), pp. 668-680.
- [17] Al-Barakati N. M., & Al-Aama, A. Y., "The effect of visualizing roles of variables on student performance in an introductory programming course," *ACM SIGCSE Bulletin*, vol. 41, pp. 228-232, 2009.
- [18] Pallant, J., (2010) *SPSS Survival Manual: A step by step guide to data analysis using SPSS*. Berkshire UK: McGraw-Hill Education.
- [19] Hundhausen, C. D., Douglas, S. A., & Stasko, A. T., (2002) "A meta-study of algorithm visualization effectiveness," *Journal of Visual Languages and Computing*, vol. 13, pp. 259-290.
- [20] Yuan, X., Vega, P., Qadah, Y., Archer, R., Yu, H., & Xu, J., (2010) "Visualization Tools for Teaching Computer Security," *ACM Transactions on Computing Education*, vol. 9, pp. 147-155.

AUTHORS

Mabroukah Amarif: received her BSc degree in Computer Science from University of Sebha, Libya, MSc in Computer Science from Universiti Sains Malaysia, and PhD in Software Engineering from Universiti Kebangsaan Malaysia. Her interests span a wide range of topics in the area of Software Engineering, Networking, Computer Security, Visual Informatic, Computer Education and programming languages. She is currently working as Assistant Professor at the departement of computer science, Faculty of Information Technology in Sebha University of Libya.



Sakeenah Ahmed: received her BSc degree in Computer Science from University of Sebha, Libya. She is currently doing her MSc in computer Sciences at Sebha University of Libya. She interests in the area of Software Engineering, Networking, Computer Security and Computer Education. she is currently working as teacher of computer sciences in secondary school of Libya



AN ANALYZING ALGORITHM BASED ON LEARNING AND SEARCHING IN CHINESE MEDICAL BIG DATA

LUO Jie, ZHOU Ziyang, SONG Qiaolin and QIU Qin'nan
*LI Zhimin

Medical Technology College, Zhejiang Chinese Medical University, Hangzhou,
China, 310053

ABSTRACT

We propose an improved analysis algorithm based on learning and searching methods in the field of Big Data to optimize TCM information management. We use TF-IDF theory in document clustering to cluster the name of Chinese Medical prescripts, construct word bag model, and combine universal hash with perfect hash, in order to establish a special key-value hashing band between Chinese Medical Data and diseases.

KEY WORDS:

TCM, Chinese Medical Massive Data, Big Data, Hashing, Index, Mapping

1. INTRODUCTION

1.1. TCM Big Data

With the dramatic development of information technologies, the memory forms of massive data generated by traditional Chinese medicine (TCM) clinical courses is transforming from paper to digital methods gradually. At the same time, the scale of TCM diagnosis and treatment data shows explosive growth. How to deal with contemporary TCM big data by information technologies has become a mainstream task. Facing with a large TCM database, manual retrieval of ancient Chinese medicine books is a very heavy work, which not only consumes a lot of time and manpower, but also easily lead to errors and omissions.

In the existing medical query system, the query system based on single table database retrieval and the single coding query system are more common. These two systems usually search the whole table index in retrieval, so the overhead of retrieval and update is extremely high when facing with mass data. Another shortage is that they can only target at one set of specific TCM data retrieval at a time, and have poor effect in data analysis and statistics, which greatly hinders TCM data mining and big data processing.

For mass data analysis and processing, there are two mainstream solutions in the field of information technology:

1) Adopt memory database and optimize compilation environment. Such schemes load relevant data into memory, thereby reducing I/O overhead. However, due to the strict database ACID

principle (atomicity, consistency, isolation, and durability), there will be unnecessary overhead and limitations for some applications with weak consistency requirements.

2) MapReduce framework[4]. The MapReduce framework consists of Map function and Reduce function. In this framework, key-value mapping is always used as a key operation to achieve good scalability and fault tolerance. But at the same time, because of the frequent I/O operations, the overhead of Map function is enormous when facing with massive amounts of data that need to be processed in real time. At present, the mainstream solutions tend to improve hardware and the schemes of memory framework. But there are few studies aiming at optimizing in the level of algorithm.

1.2 Research Process Framework

Based on the idea of machine learning, this article proposes an improved analysis algorithm to process TCM big data which is a particular but generic object. This algorithm is able to learn TCM massive data by utilizing document clustering algorithm and generate hashing weight value to map and search TCM data in hash table. In addition, we compare and evaluate the efficiency between original index model and improved model as the scale of data increasing dramatically. In the second part of this article, we introduce the hashing framework and original index model. In the third part, we propose an improved index model and. In the fourth part, we describe the details of the process of improved hashing method. In the fifth part, we compare the efficiency and overhead between two methods and evaluate them. The limitation and future improvements are discussed in the sixth part and conclusion is given in the last part.

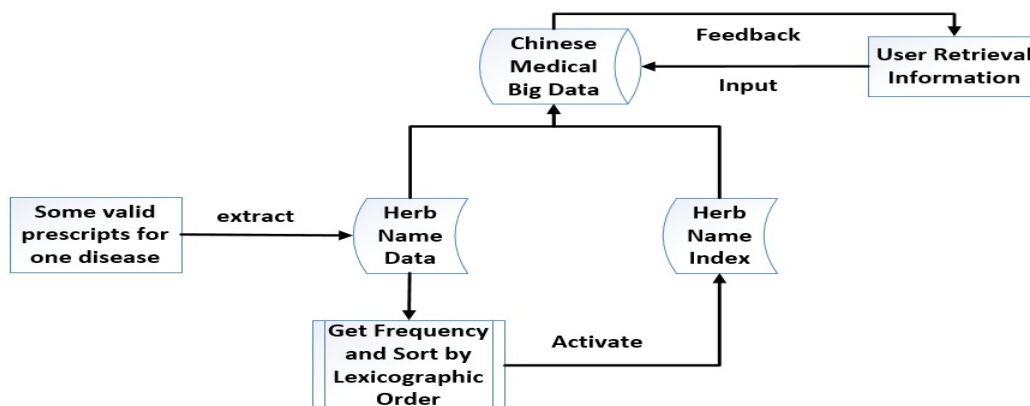
2. INDEX INITIALIZATION

Existing TCM big data includes massive traditional Chinese medicine(TCM) literature and a large number of TCM materials. And due to the complexity of TCM diagnosis and treatment, as well as a variety of TCM materials, there are tens of millions of actual clinical prescriptions for the treatment of diseases. So we select only one of many diseases and conduct a simulation experiment aiming at TCM materials and prescriptions of this disease.

2.1. Index Initialization Model

This article sort data by frequency dictionary in Universal System Index, that is, the index is generated and continuously allocated memory units according to the occurrence frequency of TCM materials.

Flow Chart 1. Index initialization model.

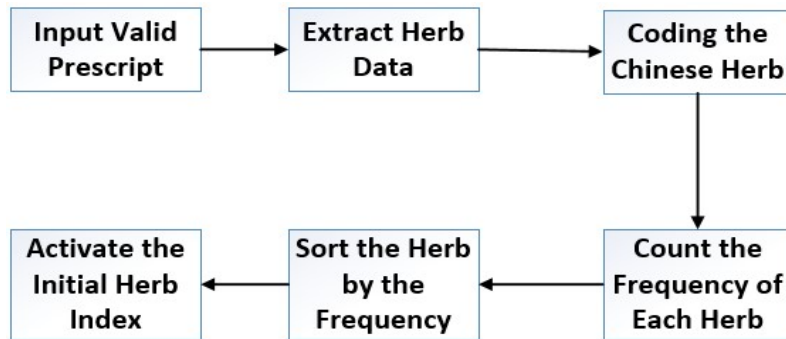


2.2. Modelling Process and Results

Each clinical prescription includes the data of TCM materials and dosages. The name of TCM materials are consisted by Chinese characters and some of them are rarely-used Chinese characters, so we code all materials by numbers and ensure each code of materials is distinct. And then the TCM materials are sorted to generate an initialization index and stored sequentially in database.

A basic index address allocation scheme of TCM database can be obtained to store massive data of TCM in order of frequency. In this way, the system can maintain and use the index to retrieve TCM massive data.

Flow Chart 2. Process of Index Initialization.



3. AN IMPROVED LEARNING MODEL

In this part, we provide an improved analyzing method to excavate TCM data. We use the same test data in this method to compare the efficiency and overhead with the original index method.

3.1. Task Object

The scale of massive TCM data not only is exponential, but will also expand with the development in the relevant fields. So the first step in this task is to design a learning system, classifying and clustering disordered TCM data to obtain scientific frequency of TCM materials. We illustrate the learning task as follows:

Task Framework 1.

Task of TCM prescription leaning
Task T: classify the materials of valid TCM prescription
Performance standard P: percentages of materials in this prescription and others
Training experience E: valid prescription data in training set

We choose one of many diseases as task object. Prescriptions for this disease vary in material dosages depending on the disease's subdivision and severity. Many doctors tend to use aliases of herbs in their prescriptions as well as omitting or elaborating the preparation methods of TCM. So we select 20 clinical prescriptions in the same category as experimental object. Then we extract and code the Chinese name and dosage of materials, generating 10 separate documents and the matrix of the name of TCM materials. In this way, a scientific TCM dictionary can be built based on document clustering.

3.2. Bag of Words - a Model of Prescription

In the modelling process, a valid prescription can be seen as a Bag of Words[18].The name of components in prescription can be seen as term vectors. In this way, we can utilize TF-IDF(term-frequency&inverse-document-frequency) to learn and analysis prescription data.

TF-IDF is a data statistics method used to analyze the importance of words in documents[19]. TF(term-frequency) refers to the frequency with which the word appears in this document. IDF(inverse-document-frequency) refers to the frequency with which the word appears in other documents[20]. The main idea of TF-IDF is: if a word appears more frequently in this paper and less frequently in other documents, it indicates that the word has a good degree of discrimination, that is, the word is more important to this document[21].

Referring to this idea, it can be considered that if a TCM material appears repeatedly in multiple prescriptions of different categories, it is unlikely to play a key role in the diagnosis and treatment of this disease, that is, it is a auxiliary ingredient or tonic maintenance type. But if the TF value of this material is large, its auxiliary proportion in such diseases can be approximately considered to be large.

We quantitatively count the dosages of TCM materials appearing in single prescription. The dosages of TCM materials are taken as the quantitative criteria of frequency to generate valid prescription document. Finally, TF-IDF frequency statistics are added into this document to construct TCM data dictionary which is shown as blow:

Table 1. TCM data dictionary (excerpts)

	P1	P2	P3	P4	P5	P6	P7	P8	P9	P10
八月扎	0	0	0	0	0	12	0	0	15	0
白芍	0	0	0	0	0	0	0	0	12	0
白术	0	0	0	0	0	0	15	0	12	0
败酱草	20	0	0	0	0	0	0	0	30	30
半枝莲	0	0	20	0	0	0	0	0	0	0
薄荷	0	9	0	0	0	0	0	0	0	0
北沙参	0	0	15	0	0	0	0	0	0	0
北秫米	0	0	0	0	0	0	0	30	30	0
蝉衣	6	9	0	0	0	0	0	0	0	0
炒稻芽	30	15	0	0	30	0	0	30	0	0
炒黄连	0	0	0	0	0	0	0	6	0	0
炒黄芩	0	15	15	0	0	0	0	0	0	0
炒竹茹	15	0	0	0	0	0	0	15	0	0
陈皮	12	12	0	0	0	12	0	0	0	0
赤芍	0	0	0	15	0	12	15	0	0	0
川朴	0	12	0	0	0	0	0	0	0	0
川朴花	12	0	0	0	9	0	0	0	0	0
淡附子	0	0	0	10	0	0	10	0	0	0

Note: In this table, 0 means the corresponding material does not appear in this prescription. And the name of TCM materials are presented in Chinese to avoid ambiguities in translation.

The TCM data dictionary can lay a foundation to the universal hashing. The TF-IDF value in this table can be calculated as follows:

Formula 1.

$$tf_{i,j} = \frac{n_{i,j}}{\sum_k n_{k,j}}$$

Formula 2.

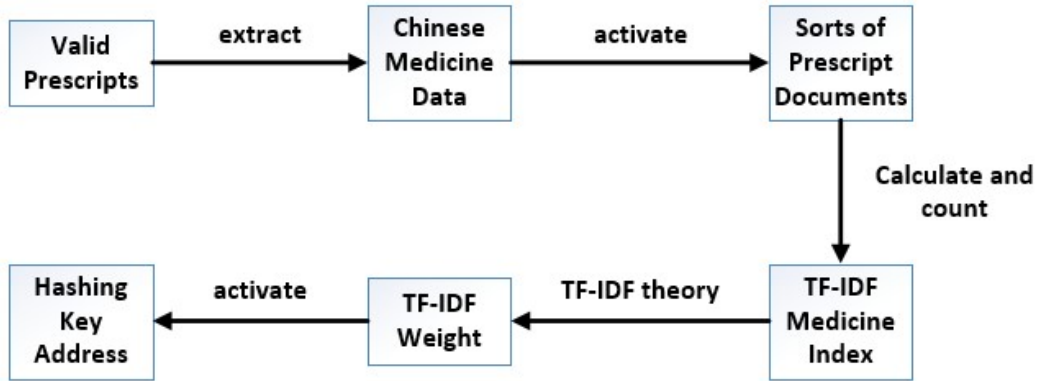
$$idf_i = \log \frac{|P|}{|\{j : t_i \in p_j\}|}$$

In formula 1, $n_{i,j}$ means the times that materials appear in prescription p_j ; $\sum_k n_{k,j}$ is the sum times that materials appear in all prescriptions. In formula 2, $|P|$ is the sum of prescriptions in this table; $|\{j : t_i \in p_j\}|$ means the number of prescriptions which contains material t_i . When a kind of material is not in a prescription, the corresponding value would be 0, which could result in a program error. So we add 1 to the denominator of the second formula to avoid that.

3.3. Algorithmic Learning Flow Chart

As we can see in this flow chart, we extract TCM prescription data and generate TF-IDF value.

Flow Chart 3. TCM data learning.



4. IMPROVED HASHING METHOD

After generating TF-IDF value, we use hashing method to further improve the algorithm next.

4.1. Universal Hashing Representation

We can infer from the TF-IDF value that, for a branch of a certain kind of disease, TCM material with high IDF value plays an important role in the prescription. TCM material with high TF value plays an auxiliary role in the treatment of the whole disease category. For the initial classification of TCM produced by TF-IDF, we need to map them into continuous memory space. The higher the TF-IDF value of TCM material, the higher the priority of its memory address. Therefore, we choose universal hashing to implement this task.

We first select a very large prime number so that every possible key values are smaller than the prime number, and then we construct a preliminary universal hash function as following:

Function 1.

$$h_{a,b}(k) = ((ak + b) \% p) \% m$$

The m in this formula above could be adjusted according to experimental results. And the cluster function of hashing is as follows:

$$\text{Function 2.} \\ H(p, m) = \{h_{a,b} : a \in Z_p^* \text{ and } b \in Z_p\}$$

In this cluster function, each Z_p can be mapped to Z_m and we can adjust the value of m to reduce the possibility of collisions in hashing.

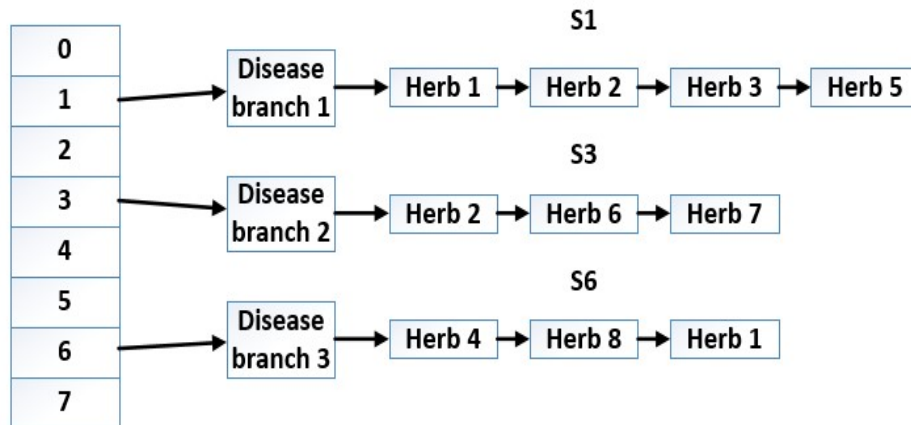
4.2. Improved Perfect Hashing

Next, we introduce the perfect hashing to improve the universal hashing. Full hashing is efficient because it has a time complexity of $O(1)$. For the massive data of TCM, we use two-level perfect hashing, and in each level, we use universal hashing. In the first level of perfect hashing, the hash function h is selected from a universal hash cluster to hash n keywords into m slots; next, in the second level of hashing, the size of the hash table is the square of the number of keys hashed into a particular slot[22] for which can reduce the conflicts in hash and can carry out the second hash depending on TF-IDF values.

With this improved perfect hashing, we use TCM material with a high weight value as the key, and put the rest of the disease information into the corresponding slot. When retrieving, we can quickly use this connection to search drug information or disease information.

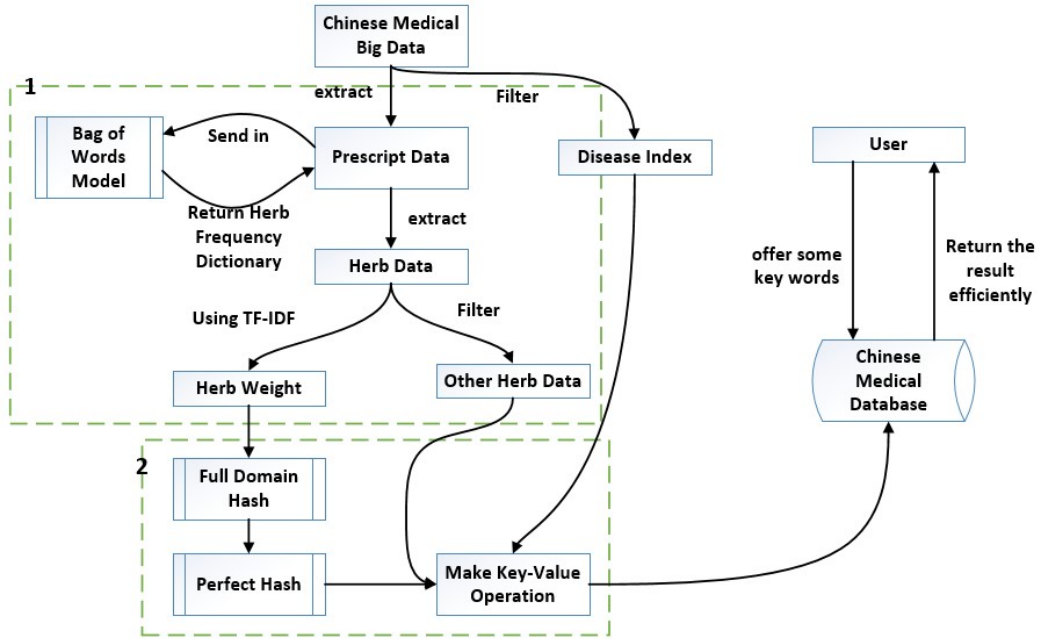
4.3. The Structure of Hashing

Model 1. memory structure of perfect hashing.



In case of conflicts, the chain address method is adopted for horizontal address expansion, in which the chain structure is in the order of weight value, so that the distance of pointer movement can be saved in querying to achieve a efficient retrieval.

The model is mainly divided into two modules: word bag processing module and hash module. In the word bag processing module, TCM materials are classified depending on prescriptions they are in and the corresponding TF-IDF values are calculated. In hash module, the data of materials, prescriptions and diseases is assigned memory addresses.



5. EVALUATION

5.1. Analysis About Index Initialization Model

Index Initialization Model is a statistical word frequency structure. When maintaining data, we chose dichotomy to insert the new data of TCM material to avoid breaking the overall alphabetical order, so the pure time complexity is $O(\log n)$. This is called pure time complexity because we also need to manipulate the amount of data inserted and other processing. Since word frequency needs to be counted and reordered, the time complexity of the whole structure[23] is shown in the following table:

Table 2. Time complexity of Index Initialization.

	Average case	Worst case
query	$O(\log n)$	$O(\log n)$
insert	$O(\log n)$	$O(\log n)$
delete	$O(\log n)$	$O(\log n)$

The time overhead of word frequency is relatively ideal, but what we should pay attention to is that this structure only construct an index. In this way, the information in the database needs to be invoked again during the query. Since each query depending on the index, the system have to switch between the index and the database to input and output. Therefore, IO overhead would be extremely large when facing with actual TCM mass data.

As for space overhead, although the space complexity of Index is $O(n)$, the scale of index would expand as the amount of data continues to grow. In this case, system have to allocate additional memory space for index, which is a large waste of space overhead.

5.2. Analysis About Improved Algorithm

The improved algorithm is divided into word bag and hashing, so we need to analyze these two parts respectively.

Firstly, the bag of word operates on matrix through the fast matrix algorithm and the idea of dimensionality reduction. The time complexity is $O(tkn)$ where t is the number of matrix iterations, k and n are the rows and columns of the matrix after dimension reduction, respectively[24]. It doesn't seem to be as efficient as index initialization, but what we should pay attention to is that, the bag of word is a training set generated from a small part of TCM data. Analyzing and summarizing the overall rules of TCM data from a small part saves a lot of overhead. In addition, the generated machine learning framework can be used for over time.

Secondly, in hash module, the time complexity depends on hash function. The time complexity of hash function is linear in text selecting, that is, the best time complexity is $O(1)$ and the worst case is $O(n)$. As for space overhead, the memory space needed is in the level of $O(n)$ because the whole hash structure can be embedded in the structure of TCM database.

Overall, combined with these two modules, the improved algorithm has lower overhead and higher efficiency when retrieving not only on time but also on space.

5.3. Initializes Index Model Evaluation

The initialization index model is established according to the occurrence frequency of TCM materials. The results of sample frequency data are shown in the figure below(In order to avoid ambiguity in translation, the names of TCM data are all in Chinese):

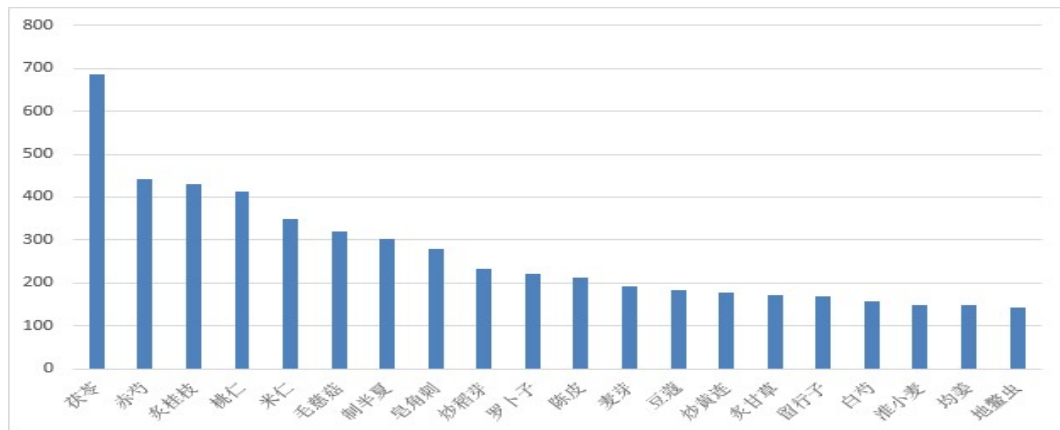


Figure 1. Part of the frequency table of TCM materials corresponding to a disease.

According to this figure, the frequency of Fuling is relatively high, but the efficacy of Fuling mainly plays a nourishing role, and the efficacy of primary treatment is not completely matched with the disease data selected in this paper. At the same time, the selection of medicinal materials is affected by many factors, such as price, efficacy, and patients' physical condition. Therefore, the initialization index method lacks rationality in the data analysis of TCM, and the logical

correlation between the data in the model structure is poor, which leads to the disorder of the logical classification and retrieval, resulting in redundant overhead.

5.4. Improved Algorithm Evaluation

In the initial learning analysis, the improved algorithm introduces tf-idf theory in the document clustering model, so that the TCM materials can be directly classified from the prescriptions and allocated with memory space continuously. So logical addresses are more centralized when retrieving so that corresponding main disease of treatment can be better excavated.

In the perfect hashing, the only possible problem is hash collision. In this case, we refer to a theorem: If n keywords are stored in a hash table and the size of this hash table m is n^2 , using a hash function H randomly selected from a universal hash function class, the probability of collision is less than $1/2$ [26]. If the hash function H is selected randomly, the collision probability of any two key words is $1/N$. If we set X is a random variable, when $m=n^2$, the expectancy of collision is as the formula below [27]:

Formula 3.

$$E[X] = \binom{n}{2} \times \frac{1}{n^2} = \frac{n^2 - n}{2} \times \frac{1}{n^2} < \frac{1}{2}$$

In this way, the collision probability of the whole perfect hash function will be less than 50%. For massive data, it can complete a better hash mapping.

5.5 Retrieval Overhead Comparison

Here we compare the time overhead between two methods, initialization index and improved perfect hashing, as the scale of data increasing.

Table 3. Experimental results.

Scale of data (10^n)	Initialization index(s)	Improved hashing(s)
1	0.187	0.186
2	0.234	0.230
3	1.026	0.503
4	1.233	0.729

As we can see from this table, improved perfect hashing method based on TF-IDF has more noticeable efficiency especially when the scale of data increasing dramatically.

6. LIMITATIONS AND IMPROVEMENTS

6.1 Limitations

TCM theory has a long history. For many difficult diseases, many regions still retain a large number of folk prescriptions, most of which lack sufficient data and theoretical support. In addition, due to the differences in patients' clinicopathological symptoms and personal physical conditions, doctors can adjust the classical prescriptions in the level of dose and material type

according to the actual situation and personal experience. Therefore, there are some difficulty in hashing and retrieval of these complex diseases and TCM data. Currently, electronic data do not account for the majority of the TCM massive data. How to unify the non-electronic data and electronic data, and how to ensure the high accuracy and efficiency of hash analysis to electronic data after it is promoted are also great challenges to the algorithm structure.

6.2 Future Improvements and Studies

We can introduce machine learning and data mining to analyze the intricate TCM semantics, names of prescriptions, diseases and materials. Through more accurate semantic analysis and clustering, we can further process data, mine and utilize its rules, aiming at the automation of TCM data processing.

7. CONCLUSIONS

This paper deeply studies the traditional Chinese medicine(TCM) massive data and propose an improved algorithm for the research and retrieval of TCM massive data. Firstly, this paper summarizes the knowledge of hashing, machine learning and other related fields, and proposes a simple initialization index algorithm to generate the processing results and take it as the control group. And then, TF-IDF theory in the field of literature clustering is used to preliminarily classify names of TCM materials and generate hash weight values. Then, the universal hash and perfect hash are used to allocate and generate key-value addresses for TCM data, so as to achieve reasonable analysis and efficient retrieval. Last but not least, the structure, logic and overhead of the two algorithms are evaluated.

Our experimental results and evaluations show that this improved algorithm is feasible in TCM data modelling and mining. It provide a method based on perfect hashing to construct TCM data dictionary which can be retrieved and managed efficiently, giving a reference to future TCM big data mining and learning. Based on this method, we will continue our research on the modernization and digitalization of TCM big data.

This thesis has done the following preparation work: 1) Foundation. Consult and study relevant data structure and data processing knowledge, and verify the correctness of the algorithm through the proof. 2) Analysis and extract data from particular experiment object, TCM prescription text, building data structure model. 3) The control experiment, through the concrete data, has carried on the actual processing to the algorithm and verified the algorithm validity. 4) Evaluation. The two algorithms are evaluated in different aspects, and the conclusion that the improved algorithm is better is obtained.

REFERENCES

- [1] RenT, XiaoX, Liu X, Sun Y. Study on knowledge acquisition of TCM prescription[J]. Chinese Journal of Information of TCM,2012, 19(7):24-27.
- [2] Zhou C. Chinese traditional medicine culture in overseas development status and reflection. World journal of integrated traditional Chinese and western medicine, 06(8), 733-733.
- [3] Li H. (2006). Characteristics and enlightenment of the development of traditional Chinese medicine in foreign countries. Chinese journal of traditional Chinese medicine, 21(6), 359-361.
- [4] J.Dean and S.Ghemawat. MapReduce, simplified data processing on large clusters. In Communications of the ACM. V.51.n.1, January 2008

- [5] http://en.wikipedia.org/wiki/Hash_table, Hash table
- [6] Ma R, Jiang H, & Zhang Q. (2008). An improved method for fast hash table lookup. *Computer engineering and science*, 30(9), 66-68.
- [7] Bertoni, G. , Daemen, J. , Michaël Peeters, & Assche, G. V. . (2014). *Sakura: A Flexible Coding for Tree Hashing*. Applied Cryptography and Network Security. Springer International Publishing.
- [8] Lu, Y. , Prabhakar, B. , & Bonomi, F. . (2006). Perfect hashing for network applications. *IEEE International Symposium on Information Theory*. IEEE.
- [9] Freitas, A. T. . (2003). Introduction to algorithms. *Resonance*, 1(9), 14-24. Page 132.
- [10] Mark L. Krotoski. Co-author, Using “Digital Fingerprints” (or Hash Values) for Investigations and Cases Involving Electronic Evidence, *United States Attorneys' Bulletin*, Vol. 62
- [11] Freitas, A. T. . (2003). Introduction to algorithms. *Resonance*, 1(9), 14-24. Page 138.
- [12] Fredman, M. L. , Fredman, M. L. , Fredman, M. L. , Fredman, M. L. , Komlos, J. , & Komlos, J. , et al. (2008). Storing a sparse table with $O(1)$ worst case access time. *23rd Annual Symposium on Foundations of Computer Science (sfcs 1982)*. IEEE.
- [13] Hagerup, T. , & Tholey, T. . (2001). Efficient minimal perfect hashing in nearly minimal space. *Symposium on Theoretical Aspects of Computer Science*. Springer-Verlag.
- [14] Fox, E. A. , Chen, Q. F. , & Heath, L. S. . (1992). A faster algorithm for constructing minimal perfect hash functions. *ACM*.
- [15] Chen kejie. (2011). Review of machine learning technology in social network analysis. *Journal of nanjing university of posts and telecommunications (natural science edition)*, 31(3), 83-89.
- [16] Klix, F. . (2010). Michalski, r. s. / carbonell, g. / mitchell, t. m. (eds.), *machine learning. an artificial intelligence approach*. *ZAMM - Journal of Applied Mathematics and Mechanics / Zeitschrift für Angewandte Mathematik und Mechanik*, 65(11), 568-568.
- [17] Han, J. . (2005). *Data Mining: Concepts and Techniques*. Morgan Kaufmann Publishers Inc.
- [18] http://en.wikipedia.org/wiki/Bag_of_words_model
- [19] Rajaraman, Anand, Ullman, Jeffrey David. *Mining of Massive Datasets: Recommendation Systems*.
- [20] <http://en.wikipedia.org/wiki/Tf-idf>, TF-IDF
- [21] http://baike.baidu.com/link?url=VYoxFr1UxhS94IGqYSmSybYQuEs4NPKcYrYng2N4ru1c1I_MUBi5BxwvDs-S15pX5PkiSacCHapHpZaH6k2Nq, Baidu Baike, tf-idf theory
- [22] Lu, Y. , Prabhakar, B. , & Bonomi, F. . (2006). Perfect hashing for network applications. *IEEE International Symposium on Information Theory*. IEEE.
- [23] http://en.wikipedia.org/wiki/Red%E2%80%93black_tree
- [24] Xu, W. , Liu, X. , & Gong, Y. . (2003). Document Clustering Based On Non-negative Matrix Factorization. *DBLP*.
- [25] C?linescu, & Gruia. (2016). 1.61-approximation for min-power strong connectivity with two power levels. *Journal of Combinatorial Optimization*, 31(1), 239-259.

- [26] Zhang min, gao xiao-hong, sun xiao-meng, xu jia-tong, li xiang-yan, & shi yanjie, et al. (2008). Pharmacological action and research progress of poria cocos. Journal of beihua university (nature), 9(1), 63-68.
- [27] Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, Clifford Stein INTRODUCTION TO ALGORITHMS, Hashing, 2006

AUTHORS

LUOJie, Master of engineering; associate professor in Medical Technology College, Zhejiang Chinese Medical University; member of computer society of China.



ZHOUIyang, Bachelor of engineering, undergraduate in Medical Technology College, Zhejiang Chinese Medical University.



SONG Qiaolin, attending doctor of traditional Chinese medicine, master's degree, research direction: clinical teaching management of traditional Chinese medicine, clinical tumor of integrated traditional Chinese and western medicine



QIU Qin'nan, Bachelor of engineering, undergraduate in Medical Technology College, Zhejiang Chinese Medical University.



Li Zhimin, associate professor, majoring in medical data processing and analysis. Corresponding author of this article.



THE IMPLICIT PATH COST OPTIMIZATION IN DIJKSTRA ALGORITHM USING HASH MAP DATA STRUCTURE

Mabroukah Amarif and Ibtusam Alashoury

Department of Computer Sciences, Sebha University, Sebha, Libya

ABSTRACT

The shortest path between two points is one of the greatest challenges facing the researchers nowadays. There are many algorithms and mechanisms that are designed and still all according to the certain approach and adopted structural. The most famous and widely used algorithm is Dijkstra algorithm, which is characterized by finding the shortest path between two points through graph data structure. It's obvious to find the implicit path from the solution path; but the searching time varies according to the type of data structure used to store the solution path. This paper improves the development of Dijkstra algorithm using linked hash map data structure for storing the produced solution shortest path, and then investigates the subsequent implicit paths within this data structure. The result show that the searching time through the given data structure is much better than restart the algorithm again to search for the same path.

KEYWORDS

Dijkstra algorithm, data structure, linked hash map, time complexity, implicit path, graph

1. INTRODUCTION

The graph is defined as a set of nodes (Vertices) and edges that link these nodes. Each edge is marked with a weight value describing the cost between the connected nodes. There are two types of graph; directed graph for which each node is directed by one way to any other node, and undirected graph for which its possible to go to and back from the same way between two connected nodes. The issue of the shortest path problem is related to graph theory, which is one of the most important topics for researchers [1, 2, 3, 4, 5, 6, 7]. It concerns with finding the shortest path between two nodes, or between a node as a source to all other nodes, depending on the weights of the edges that link these nodes [8]. The graph theory and shortest path are used widely especially in the practical applications of various fields, the most important are roads, transport networks and geographic information systems [9, 10, 11, 12].

Various algorithms have been created to find the shortest path within graphs. These algorithms depend on the graph and path type. The most famous of these algorithms are Dijkstra algorithm [13], Bellman, the Johnson, and the Floyd Warshall algorithms [14].

Based on the rapid development and handling of the huge amount of road networks data and their development methods, researchers try to improve the previous algorithms or develop their own [4, 7, 9, 10, 11, 12]. They are also exploring more various ways and methods to get the lowest cost of path into a large number of nodes and speeding up their search process. Some of them are working on reconstructing the graphs in an attempt to reduce the cost of searching time either by

compression, optimization or subgraphs [1, 2, 3, 4, 5, 6, 15]. Others are replacing data structure with another [18, 19, 20, 21].

However, the field remains open to researchers to discover more ways and mechanisms in an attempt to get the lowest search cost within a large number of nodes and accelerate the search process which is a challenge until this time. This paper proposes an improvement of Dijkstra algorithm using a special data structure (linked hash map) for storing the solution path (shortest path given by Dijkstra algorithm) to optimize the searching time for the implicit paths. The proposed algorithm uses Dijkstra algorithm with priority queue implemented by min heap to find the solution path (shortest path). The solution path is stored into data structure of array list contains of a linked hash map elements. We have tested the proposed algorithm with different graphs sizes up to 10000 nodes. The following section describes the most related works to our paper while section 3 explains the algorithm description. Analysis and results are described in section 4. A discussion of this paper is explained in section 5 and the conclusion is provided in section 6.

2. RELATED WORK

Dijkstra algorithm was designed by the Dutch computer scientist Edsger Dijkstra in 1956 and published in 1959 [13]. It's the most popular algorithm in the area of finding shortest path from single source to a node destination, or multiple nodes destination within a graph [16, 17, 22]. It can also be used to find the shortest route costs from the source to the destination by stopping the algorithm once the shortest route is set to the specific target.

This algorithm has been considered by many researchers to improve the shortest path cost by minimizing the searching time (time complexity) using different data structure. Jain et. al. improve the Dijkstra algorithm by using priority queue and linked list [18]. It has been noticed that by using a graph represented by thier adjacncey lists and the priority queue implemented as a min-heap, the time efficiency is in $O(|E| \log |V|)$, where V is the number of nodes and E is the number of edges which connected these nodes. if the priority queue is implemented using an advanced data structure called the Fibonacci heap, the time becomes $O(|V| \log V+E)$, and its imporved [16].

Time efficiency could be imporved by exploit the solution path to get the implicit paths if they are queried again. From the literature, most algorithms may provide this feature, but there is no such explanation or imporvement of it. A suitable data structure could improve the time efficiency for the whole algorithm if used propably for storing the solution path and then search for the implicit path. In this paper, we propose for an improved algorithm based on Dijkstra algorithm by using a special data structure (linked hash map) to store the solution path. The proposed algorithm is tested with different number of nodes. Results are recorded to justify the Algorithm validity.

3. THE PROPOSED ALGORITHM DESCRIPTION

The idea of Dijkstra algorithm is genius and amazing. In addition, it's simple and easy to understand especially by using non complicated data structure such as priority queue implemented as a min-heap. Although Fibonacci heap has achieved greater success and better performance, however, it's complicated and often theoretical more than practical issues [16]. Based on all of that, we use Dijkstra algorithm with priority queue implemented as a min-heap. For storing the solution path, we use a linked hash map within array list. This can give us a fast searching time equal to $O(1)$ as an efficiency time. The following steps explain the main idea of Dijkstra algorithm [16].

```

DIJKSTRA. (G,w,s)
1 INITIALIZE-SINGLE-SOURCE.(G, s)
2 S=∅;
3 Q=G.V
4 while Q ≠ ∅ ;
5 u=EXTRACT-MIN(Q)
6 S=S∪{u}
7 foreach vertex v ∈G.Adj[u]
8 RELAX (u,v,w)

```

in step 1, the initialization of the source node s in the graph G is carried out. Step 2 initializes the set S to the empty set. The algorithm maintains the invariant that $Q = V - S$ at the start of each iteration of the while loop of step 4 until step 8. Step 3 initializes the min-priority queue Q to contain all the vertices in V ; since $S = \emptyset$ at that time, the invariant is true after step 3. Each time through the while loop of steps 4 until step 8, step 5 extracts a vertex u from $Q = V - S$ and step 6 adds it to set S , for the first time through this loop, $u = s$. Vertex u , therefore, has the smallest shortest path estimate of any vertex in $V - S$. Then, step 7 and step 8 relax each edge (u,v) . Notice that w is the weight of the given edge.

We have built the storage data structure and identified a variant of the type of array-list which contains a set of linked-hash-map (each value within the array-list is actually linked-hash-map includes a particular path) to store any path that has been queried. The hash-map is used to store the nodes and the distance between them. Each hash-map consists of (Key, Value). Each node and adjacent is stored in the key, while the distance between them is stored in Value. The following code describes the idea of storing and searching operations of the storage data structure.

//storing operation

```

public static ArrayList<LinkedHashMap<Integer,Integer>> AddtoPaths(List<Integer> path,int
srcc,int destt)
{
    if(path!=null)
    {
        if(AddHashMap.size()!=0)
            AddHashMap.clear();
        for (int i = 0; i < path.size() ; i++)
            { AddHashMap.put(path.get(i),i); }
        IndexHashMap=(LinkedHashMap) AddHashMap.clone();
        AddHashMap.clear();
        if(!dublicateData(IndexHashMap,paths1,srcc,destt))
        {
            paths1.add(IndexHashMap);
            Addtolinked1(IndexHashMap,paths1);
        }
    }
    return paths1;
}

```

```

public static boolean dublicateData(LinkedHashMap<Integer,
Integer>IndexHashMap1,ArrayList<LinkedHashMap<Integer, Integer>> paths11,int srcc1,int destt1){
    for(LinkedHashMap<Integer, Integer> p:paths11){
        if(p.equals(IndexHashMap1)){
            return true;
        }
    }
}

```

```

    }
  }
  for(LinkedHashMap<Integer, Integer> s:paths11)
  {
    if( s.containsKey(srcc1) && s.containsKey(destt1) )
    { return true; }
  }
  return false;
}

public static void Addtolinked1(LinkedHashMap<Integer,Integer>
path,ArrayList<LinkedHashMap<Integer,Integer>> paths1){
  int ii=1;
  int i;
  if(paths1.size()>0 ){
  i=paths1.size()-1;
  }else{i=paths1.size();}
  if(path!=null){
    comp = new ArrayList<Integer>(path.keySet());
    for (int j = 0; j < comp.size() ; j++) {
if(Number_of_Nodes.containsKey(comp.get(j)) && Number_of_Nodes.get(comp.get(j))!=null
&& Number_of_Nodes.get(comp.get(j)).size()>0)
{
  Number_of_Nodes.get(comp.get(j)).add(i);
  }else{
  Number_of_Nodes.put(comp.get(j),new ArrayList<Integer>());
  Number_of_Nodes.get(comp.get(j)).add(i);
  }
  } comp.clear();
} }
} }

//searching operation
public static ArrayList findShortestPathsDS(int src,int dest,Graph graph)
{
  start_time2 = System.nanoTime();
  ArrayList subPath = new ArrayList();
  Number_of_Nodes
  From=Number_of_Nodes.get(src);
  To=Number_of_Nodes.get(dest);
  if(MainClass.From!=null && MainClass.To!=null && MainClass.From.size()>0 &&
MainClass.To.size()>0)
  {
    SO=FoundContains(From,To);
  } else
  {
    time11=0.0;
    text = "\n The path is not implicit, So The path will be calculated using Dijkstra";
    subPath= (ArrayList)graph.findShortestPaths(src, dest);
    end_time2 = System.nanoTime();
    difference2 = (end_time2 - start_time2) / 1e6;
    time11=Graph.time;
    return subPath;
  }
  if(SO >=0)
  {

```

```

        if( (paths1.get(SO).containsKey(src)) && (paths1.get(SO).containsKey(dest)) )
        {
            found=true;
            i=SO;
        }
    if(found)
    {
        src1=paths1.get(SO).get(src);
        dest1=paths1.get(SO).get(dest);
        ss1=new ArrayList(paths1.get(SO).keySet());
        if (src1 <= dest1)
        {
            for (ii = src1; ii <= dest1; ii++ )
            { subPath.add(ss1.get(ii)); }
        }else{
            for (ii = src1; ii >= dest1; ii-- )
            {
                if (ii >=0)
                subPath.add(ss1.get(ii));
            }
        }
        end_time2 = System.nanoTime();
        difference2 = (end_time2 - start_time2) / 1e6;
    } else{
        time11=0.0;
        text = "\n The path is not implicit,So The path will be calculated using Dijkstra";
        subPath= (ArrayList)graph.findShortestPaths(src, dest);
        end_time2 = System.nanoTime();
        difference2 = (end_time2 - start_time2) / 1e6;
        time11=Graph.time;
    }
    return subPath;
} else{
    time11=0.0;
    text = "\n The path is not implicit,So The path will be calculated using Dijkstra";
    subPath= (ArrayList)graph.findShortestPaths(src, dest);
    end_time2 = System.nanoTime();
    difference2 = (end_time2 - start_time2) / 1e6;
    time11=Graph.time;
}
}
return subPath;
}
}
public static Integer FoundContains(ArrayList<Integer> From11,ArrayList<Integer> To11)
{
    int SO1=-1;
    To12.clear();
    From12.clear();
    if(From11.size()>0 && To11.size()>0)
    {
        if(From11.size()>=To11.size())
        {

```

```

for (int i = 0; i < To11.size(); i++)
    { To12.put(To11.get(i), i); }
for (int i = 0; i < From11.size(); i++)
    {
        if(To12.containsKey(From11.get(i)))
            {
                SO1=From11.get(i);
                break;
            }
    }
} else {
    for (int i = 0; i < From11.size(); i++)
        { From12.put(From11.get(i), i); }
    for (int i = 0; i < To11.size(); i++)
        {
            if (From12.containsKey(To11.get(i)))
                {
                    SO1=To11.get(i) ;
                    break;
                }
        }
    } return SO1;
} return SO1; }

```

4. THE ALGORITHM ANALYSIS AND RESULTS

The time complexity of hash-map, linked-hash-map and array-list are different according to the kind of operations. Table 1 and 2 describes the time for each given data structure.

Table 1. The time complexity of hash-map and linked-hash-map

	get	Contains Key	next	note
Hash-map	O(1)	O(1)	O(h/n)	h is the table capacity
Linked-hash-map	O(1)	O(1)	O(1)	

Table 2. The time complexity of array-list

	get	add	contains	next	remove	Iterator remove
Array-list	O(1)	O(1)	O(n)	O(1)	O(n)	O(n)

According to the previous tables, n means the number of nodes (V), and m means the number of edges (E). The total time for putting the new solution path in the given data structure is:

$$O(n \log n) + O(n) + O(n \log n) = O(n) + O(2n \log n) = O(2n \log n) = O(n \log n)$$

In the other side, the total time complexity of the query from the data structure is:

$$O(\log n) + O(\log n) + O(n \log n) + O(\log n) + O(\log n)$$

$$= O(4 \log n) + O(n \log n)$$

$$= O(\log n) + O(n \log n)$$

$$= O((1+n) \log n) = O(n \log n)$$

We run the original Dijkstra algorithm and the proposed algorithm with directed graph; where there is only one way from a node to other. We also run both of them with different number of

nodes in order to get the shortest path between a given source node and destination. After the solution path is found, it's stored in the given data structure (array-list of linked-hash-map). Then, an inquiry for the implicit path is taken place. If the implicit path is found within the stored solution path in the given data structure, the time is calculated and recorded, else; the searching using the original Dijkstra algorithm is started again and the total time is recorded. The averages of each recorded times are calculated. These operations are repeated many times with different number of nodes. Table 3 and 4 contain the averages values of the time when the required path is the implicit path of the stored solution path and figure 1 and 2 shows the results analysis.

Table 3. The run time of the implicit path (directed graph)

No. of nodes	Dijkstra using only min-heap Time average	Dijkstra using min-heap with ArrayList<LinkedHashMap> Time average
100	0.66256	0.28489
200	1.47381	0.3076
300	2.06859	0.34078
400	4.15724	0.38133
500	5.56397	0.28971
600	2.552	0.24657
700	3.15004	0.51842
800	8.35776	0.22292
900	5.11631	0.21436
1000	4.81021	0.2146

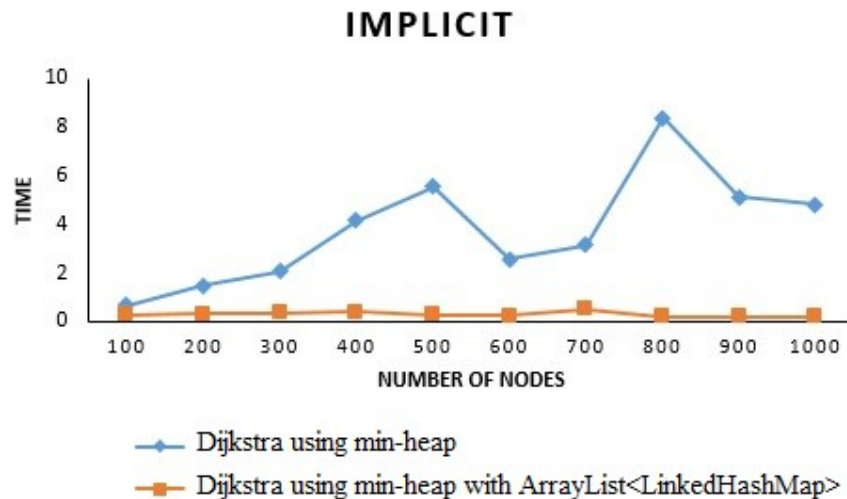


Figure 1. The run time of 100-1000 number of nodes (directed graph)

From figure 1, we notice that the time decreases as the number of nodes increases. Time average is equal to 0.28489 when the number of nodes is 100, then time increases with unobserved amount. It gives the highest value when 700 nodes and starts to decrease at 800 nodes and above.

Table 4. The run time of the implicit path (directed graph)

No. of nodes	Dijkstra using only min-heap Time average	Dijkstra using min-heap with ArrayList<LinkedHashMap> Time average
1000	4.81021	0.2146
2000	11.03552	0.18769
3000	13.76008	0.17272
4000	12.66174	0.19816
5000	15.29122	0.16887
6000	20.46555	0.18637
7000	47.87112	0.1797
8000	27.05117	0.17704
9000	55.55694	0.18778
10000	42.26821	0.17102

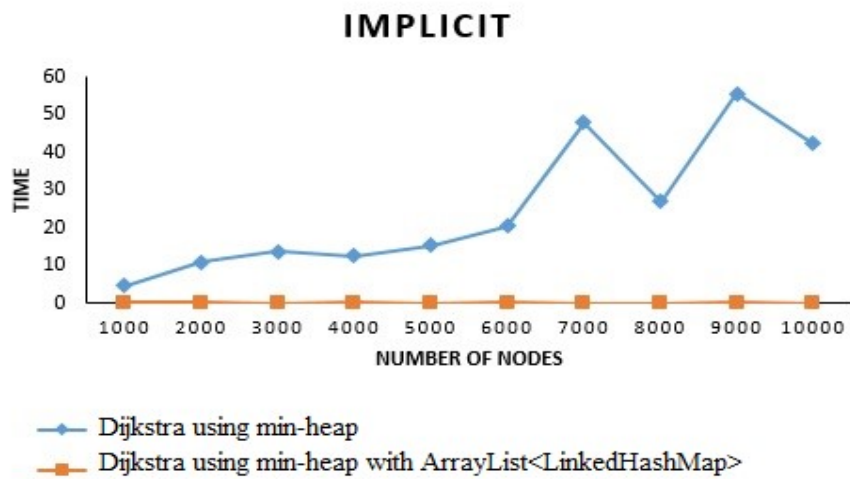


Figure 2. The run time of 1000-10000 numbers of nodes (directed graph)

Looking at figure 2, nodes numbers are various from 1000 to 10000 nodes. We realize that the time decreases as the number of nodes increase and that improved the proposed algorithm validity. We observe a linear time for the given data structure; ArrayList<LinkedHashMap>.

Table 5 and 6 contain the averages values of the time when the required path is not the implicit path of the stored solution path and figure 3 and 4 show the results analysis.

Table 5. The run time of the non-implicit path (directed graph)

No. of nodes	Dijkstra using by min-heap	Dijkstra using by min-heap with ArrayList<LinkedHashMap>
100	0.86681	0.92647
200	2.30324	2.41939
300	3.15914	3.27483
400	3.45716	3.54365
500	4.5522	4.64095
600	4.08183	4.17757
700	7.1498	7.26413
800	5.8222	5.90921
900	7.55844	7.65247
1000	7.73631	7.84603

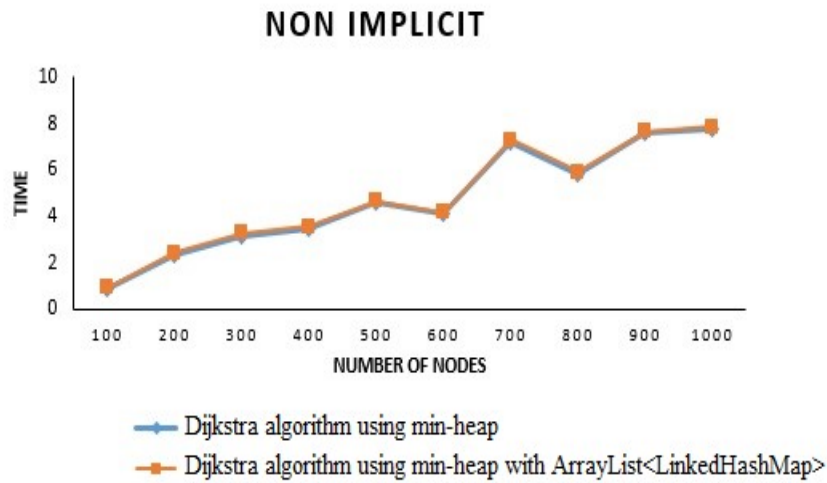


Figure 3. The run time of 100-1000 number of nodes (directed graph)

Figure 3 describes the run time of non-implicit path if it's not included within the stored solution shortest path. The relationship between run time and nodes numbers is shown. It is observed that the time for the proposed algorithm is almost equal to the original Dijkstra algorithm with min-heap. There is a slightly fixed difference as the nodes numbers increases.

Table 6. The run time of the non-implicit path (directed graph)

No. of nodes	Dijkstra using by min heap	Dijkstra using by min heap with ArrayList<LinkedHashMap>
1000	7.73631	7.84603
2000	11.90012	12.02769
3000	17.38345	17.50731
4000	31.68681	31.79614
5000	35.59574	35.68865
6000	57.20521	57.31768
7000	67.07719	67.16318
8000	35.34394	35.4589
9000	73.9846	74.109
10000	69.43308	69.54271

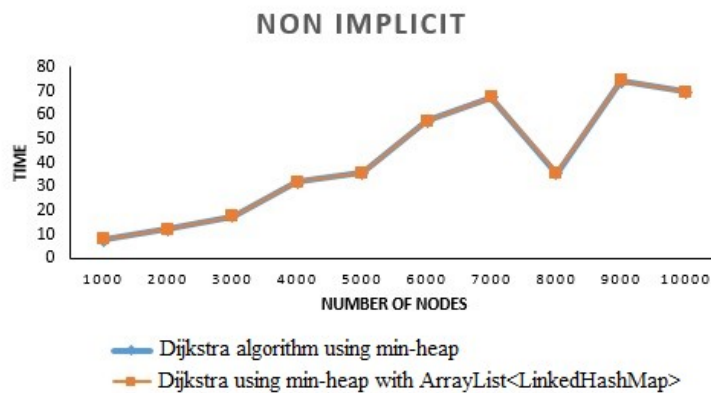


Figure 4. The run time of 1000-10000 number of nodes (directed graph)

From figure 3 and 4, we notice that almost the same time is observed for searching for a path which is not a sub path from the solution shortest path.

5. DISCUSSION

In this paper, a case study of random directed graph with different number of nodes has been carried out in order to test the validity of our proposed algorithm. The same case study has been given to the original Dijkstra algorithm with priority queue implemented as a min-heap. The results of both algorithms have been recorded and analysed. Comparisons between these results have shown that the proposed algorithm is almost the best. Although more data structures have been used within the proposed algorithm, however, the enlarged storage is available for all of the current devices, even for the smallest one. We argue that data storages aren't problem if the performance of the given algorithm is higher and success. It has been observed that the searching time of the original algorithm is almost the same as the time of the proposed algorithm if the required path is not an implicit path within the solution path. Regarding of the results in the literature, run time is various depending on the type of used data structure and also CPU speed. Although our results give a good time average for searching within a solution path, we realize that the greater the number of nodes, time becomes less and less. This can work in a large storage of nodes especially in a huge road network.

6. CONCLUSION AND FUTURE WORK

The analysis of searching time for implicit path seems to be rare and less concern. Most available algorithms expect the nature of easy time searching for sub path within the solution path. Well, data structure plays main role in the whole procedures and operations. We realize this point when we start proposing our idea for improvement of the available Dijkstra algorithm. We consider improving the time complexity of implicit path within the solution path as a first starting point. According to our results, the improvement of the proposed algorithm is achieved with regard of directed graph. This type of graph could reflect the one way road in reality. The proposed algorithm achieves the best result especially for a large number of nodes. Our future work will consider the undirected graph. We also plan to apply our proposed algorithm to a real network road map to show the performance and improve the validity from a real point of view.

REFERENCES

- [1] Arman, N., & Khamayseh, F., (2015) "A Path-Compression Approach for Improving Shortest-Path Algorithms," *International Journal of Electrical and Computer Engineering*, vol. 5, p. 772,.
- [2] Broumi, S., Bakal, A., Talea, M., Smarandache, F., & Vladareanu, L., "Applying Dijkstra algorithm for solving neutrosophic shortest path problem," *International Conference on Advanced Mechatronic Systems (ICAMechS)*, 2016, pp. 412-416.
- [3] Gao, J., Zhao, Q., Ren, W., Swami, A., Ramanathan, R., & Bar-Noy, A., (2015) "Dynamic shortest path algorithms for hypergraphs," *IEEE/ACM Transactions on Networking*, vol. 23, pp. 1805-1817.
- [4] Gutenschwager, K., Völker, S., Radtke, A., & Zeller, G., "The shortest path: Comparison of different approaches and implementations for the automatic routing of vehicles," in *Simulation Conference (WSC)*, Proceedings of the 2012 Winter, 2012, pp. 1-12.
- [5] Khamayseh, F., & Arman, N., (2015) "Improvement of Shortest-Path Algorithms Using Subgraph's Heuristics". *Journal of Theoretical & Applied Information Technology*, vol. 76.

- [6] Niemeyer, K., E., & Sung, C.-J., (2016) "On the importance of graph search algorithms for DRGEP-based mechanism reduction methods," *Combustion and Flame*, vol. 158, pp. 1439-1443.
- [7] Shu-Xi, W., (2012) "The improved dijkstra's shortest path algorithm and its application," *Procedia Engineering*, vol. 29, pp. 1186-1190.
- [8] Douglas, W., B., (2001) *Introduction to Graph Theory*, Pernice Hall.
- [9] Chandra, "Shortest Path Problem for Public Transportation Using GPS and Map Service," 2012.
- [10] Saab, Y., & VanPutte, M., (1999) "Shortest path planning on topographical maps," *IEEE Transactions on Systems, Man, and Cybernetics-Part A: Systems and Humans*, vol. 29, pp. 139-150.
- [11] Xing, S., & Shahabi, C., "Scalable shortest paths browsing on land surface," in *Proceedings of the 18th SIGSPATIAL International Conference on Advances in Geographic Information Systems*, 2010, pp. 89-98.
- [12] Faro, A., & Giordano, D., (2016) "Algorithms to find shortest and alternative paths in free flow and congested traffic regimes," *Transportation Research Part C: Emerging Technologies*, vol. 73, pp. 1-29.
- [13] Dijkstra, E. W., (1959) "A note on Two Problems in Connexion with graphs", *Numerische Mathematik*, 1, 269-271.
- [14] Thorat, S., & Rahane, S., (2016) "Review of Shortest Path Algorithm", *IRJET*, vol 3, issue 8.
- [15] Yao, B., Yin, J., Zhou, H., & Wu, W., (2016) "Path Optimization Algorithms Based on Graph Theory," *International Journal of Grid and Distributed Computing*, vol. 9, pp. 137-148.
- [16] Cormen, T., Leiserson, C., Rivest, R., & Stein, C., (2009) *Introduction to Algorithms*, 3rd ed., MIT Press, London.
- [17] Levitin, A., (2012) *Introduction to the Design and Analysis of Algorithms*, 3rd ed., Pearson Education, Inc., Addison-Wesley.
- [18] Jain, A., Datta, U., & Joshi, N., (2016) "Implemented modification in Dijkstra's Algorithm to find the shortest path for N nodes with constraint," *International Journal of Scientific Engineering and Applied Science*, vol. 2, pp. 420-426.
- [19] Xie, D., Zhu, H., Yan, L., Yuan, S., & Zhang, J., "An improved Dijkstra algorithm in GIS application," in *World Automation Congress (WAC)*, 2012, pp. 167-169.
- [20] Lu, J., & Dong, C., "Research of shortest path algorithm based on the data structure," in the *3rd International Conference of Software Engineering and Service Science (ICSESS)*, 2012 IEEE, 2012, pp. 108-110.
- [21] Kong, D., Liang, Y., Ma, X., & Zhang, L., "Improvement and Realization of Dijkstra Algorithm in GIS of Depot," in the *International Conference on Control, Automation and Systems Engineering (CASE)*, 2011, 2011, pp. 1-4.
- [22] Deng, Y., Chen, Y., Zhang, Y., & Mahadevan, S., (2012) "Fuzzy Dijkstra algorithm for shortest path problem under uncertain environment," *Applied Soft Computing*, vol. 12, pp. 1231-1237.

AUTHORS

Mabroukah Amarif: received her BSc degree in Computer Science from University of Sebha, Libya, MSc in Computer Science from Universiti Sains Malaysia, and PhD in Software Engineering from Universiti Kebangsaan Malaysia. Her interests span a wide range of topics in the area of Software Engineering, Networking, Computer Security, Visual Informatic, Computer Education and programming languages. She is currently working as Assistant Professor at the departement of computer science, Faculty of Information Technology in Sebha University of Libya.



Ibtusam Alashoury: received her BSc degree in Computer Science from University of Sebha, Libya. She is currently doing her MSc in computer Sciences at Sebha University of Libya. She interests in the area of Software Engineering, System Analysis and Web design. she is currently working as a technical Engineer in the information development project of Sebha University of Libya.



IN-VEHICLE CAMERA IMAGES PREDICTION BY GENERATIVE ADVERSARIAL NETWORK

Junta Watanabe and Tad Gonsalves

Department of Information & Communication Sciences,
Faculty of Science & Technology
Sophia University, Tokyo, Japan

ABSTRACT

Moving object detection is one of the fundamental technologies necessary to realize autonomous driving. In this study, we propose the prediction of an in-vehicle camera image by Generative Adversarial Network (GAN). From the past images input to the system, it predicts the future images at the output. By predicting the motion of a moving object, it can predict the destination of the moving object. The proposed model can predict the motion of moving objects such as cars, bicycles, and pedestrians.

KEYWORDS

Deep Learning, Image Processing, Convolutional Neural Network, GAN, DGAN

1. INTRODUCTION

Moving object detection is one of the fundamental technologies necessary to realize autonomous driving. Object detection has received much attention in recent years and many methods [1- 5] have been proposed. However, most of them are not enough to realize moving object detection, chiefly because these technologies perform object detection from one single image. Therefore, they can get the position of the desired object, but they cannot discriminate whether the object is stationary or non-stationary. For the same reason, they cannot predict the future positions of the object. This kind of technology is therefore not apt for autonomous driving. Normally, there are many pedestrians on a sidewalk. Autonomous cars driving on a roadway must discriminate whether the pedestrians will continue to be on the sidewalk or get into the roadway at a given point in time in the future.

Multiple ways to solve this problem have been considered - predicting future positions of the objects from their past positions, and predicting future camera images from past camera images, for instance. However, the former has a disadvantage. Processing time is directly proportional to the number of objects in a camera image. If there are many objects in a camera image, processing time may be too long to realize autonomous driving. Conversely, the latter is not affected by the number of objects in the image. It does not count the objects in a camera image when it predicts the future. It predicts the future positions of the objects at a time as a future image. Then, precise object detection methods detect the future positions of the object from the image of prediction. Hence, it is assumed that the latter outperforms the former.

It is not enough to predict a future image by using just a single image. This is because we cannot guess the speed of objects in an image. For example, when there are two cars in an image, we cannot guess if they are stationary or not. We guess the speed of the cars from their past and current positions. Therefore, the future image prediction needs some past images.

Multiple ways to predict future images from past images are considered, too. This paper suggests a model which predicts future images with Generative Adversarial Networks (GANs) [6]. GANs consist of two models: a generative model and a discriminative model. GANs learn a loss that discriminate whether the output image is real or fake, while training a generative model to minimize a loss. The image generated by Deep Convolutional GANs (DCGANs) [7] is often clear. It is because blurry images which look fake will not be tolerated by a discriminative model. To get clear prediction image, the method proposed in this paper uses GANs.

2. RELATED WORK

Generative Adversarial Networks Generative adversarial networks are generative models. The purpose of GANs is to generate a fake data which is indistinguishable from a real one. GANs consists of two models: a generator G and a discriminator D . G generates a fake data $G(z)$ from a noise z . D discriminates whether an input data x is real or fake. They are adversarial. Therefore, G will generate a fake data which is indistinguishable from a real one.

Pix2pix: This model can solve the image-to-image translation. A generator of pix2pix [8] uses a “U-Net” based architecture. The U-net [9] is a U-shaped convolutional neural network. The shape allows the U-net to make an output image which has features extracted in shallow layers and deep layers.

MoCoGAN: Motion and Content decomposed Generative Adversarial Network (MoCoGAN) [10] is a network which generates a video. MoCoGAN uses two discriminators: an image discriminator and a video discriminator. The image discriminator discriminates whether an input image is real or fake. The video discriminator discriminates whether an input video is real or fake.

ResNet: Residual Network (ResNet) [11] makes it easy to train deep neural networks. In ResNet, layers are reformulated as learning residual functions with reference to the layer inputs. When training deep networks, one often runs into the degradation problem which is prevented by the use of ResNet.

The approaches of moving object detection by cameras are [12], [13], [14]. They indicate how to detect moving objects from camera images. Our approach is how to predict future camera images. Therefore, we can predict positions of future moving objects by combining their methods and our method.

Our proposed method only uses an in-vehicle camera. Methods which use multiple sensor are [15], [16], [17]. Sensors enable measurement of the distance between a vehicle and a moving object. However, it is difficult to install sensors. Our method only uses a camera and hence the implementation is easier and cost-effective.

3. METHODS

Our generator predicts 4 future images from 3 past images. All these images are different from one another. For example, 1 second later, 2 seconds later, 3 seconds later and 4 seconds later. It seems the generator should generate one future image, not 4 future images. This is because if the generator repeats generating 1 future image, we will get more future images. The reason for predicting 4 future images is that training the generator that predicts 4 future images is easier than training the generator that predicts 1 future image. In the early stages, the generators generate blurry images. If we use the generator that predicts 1 future image, the generator predicts an image which is 2 seconds later from a blurry image which is 1 second later. The generator predicts an image which is 3 seconds later from a more blurred image which is 2 seconds later.

The image which is 3 seconds later is too different from the ground truth. In training the generator, we calculate a loss between the image and the ground truth. However, the image is predicted by the generator from the blurry image. Calculating the loss between the image and the ground truth cannot be said to be the right thing. Therefore, the generator calculates not 1 picture but multiple pictures. The reason for predicting 4 pictures has to do with the capacity of the Graphic Processing Units (GPUs). If we make the generator predict 4 pictures, the datasets will exceed the capacity of my GPUs. Nevertheless, if we make the generator to predict a few pictures, a discriminator of videos will not work. It is because the task of the discriminator is to discriminate whether an input video is real or fake. The shorter the videos are, the closer it is to discriminate still images from video images. Therefore, if we want the generator to generate superior videos, we must make the video long. The reason for predicting images from 3 images is the capacity of Graphic Processing Units (GPUs), too.

3.1 Network Architectures

Our network consists of 3 sub-networks: a generator G , an image discriminator D_i and a video discriminator D_v (Figure 1). The generator G is a generative model that learns a mapping from an input video clip x to an output video clip V . We define a predicted video clip as PV . The function of generator G can be expressed as:

$$\begin{aligned} PV &= G(x) \\ &= \{PI_{t+3}, PI_{t+4}, PI_{t+5}, PI_{t+6}\} \end{aligned} \quad (1)$$

$$x = \{I_t, I_{t+1}, I_{t+2}\} \quad (2)$$

$$V = \{I_{t+3}, I_{t+4}, I_{t+5}, I_{t+6}\} \quad (3)$$

PI means predicted image. D_i and D_v come from MoCoGAN.

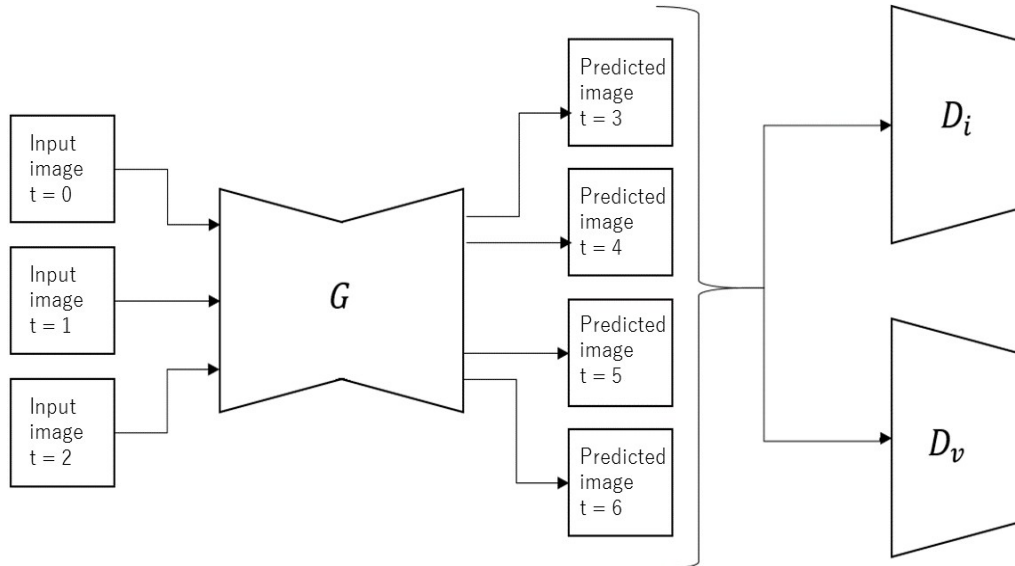


Figure 1. Our proposed method

3.1.1 Generator

The generator consists of 4 parts: pre-processor, encoder, concatenator and decoder.

Preprocessor: Preprocessor has a convolutional layer (3×3). The size of the input image is $128 \times 128 \times 3$. The layer resizes it to $128 \times 128 \times 16$. After the convolution, we do Batch Normalization (BN) [18] to speed up the training process.

Encoder: The input is fed to the ResBlock of the Encoder and Average Pooling is repeated. ResBlock consists of 4 residual units (ResUnit). Each of the ResUnit consists of 6 layers and shortcut connections. This structure is found in [19]. The Encoder successively outputs the images after passing through the ResBlock. In all, here are 3 Encoders, corresponding to the $t = 0, 2, 3$ output.

Concatenator: Concatenator connects the same-sized outputs from the Encoder and passes them through the Res-Block. The number of the channels in the ResBlock output is $4/3$ times that of the input. This is because there are 4 Decoders corresponding to 3 Encoders.

Decoder: There are 4 Decoders, handling the $t = 3, 4, 5, 6$ outputs. The output array from the Concatenator is split into 4 parts, each of which is then fed to the Decoder. The $8 \times 8 \times 256$ convolutions performed at the Encoder are then subjected to deconvolution [20] inside the Decoder.

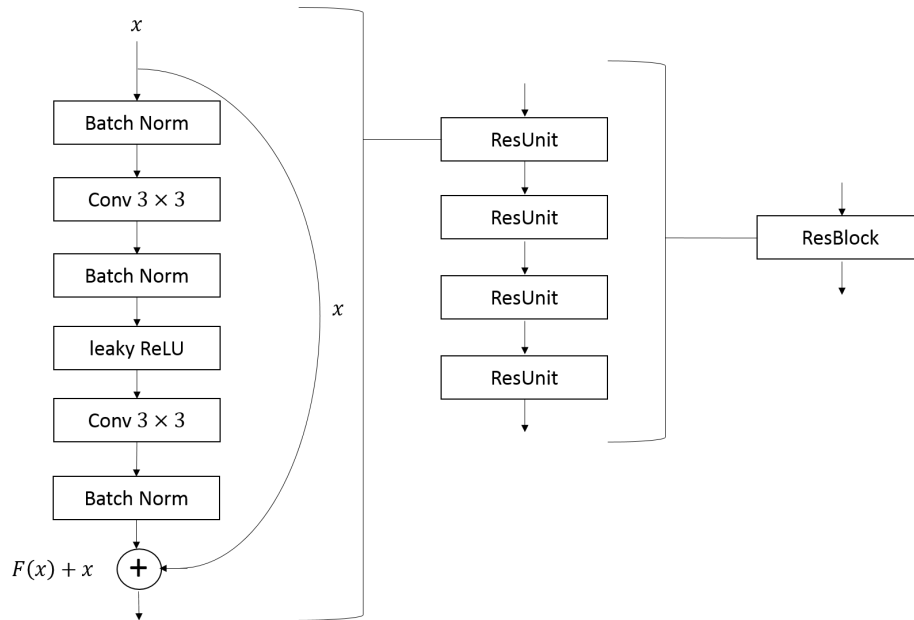


Figure 2. ResBlock of PyramidNet

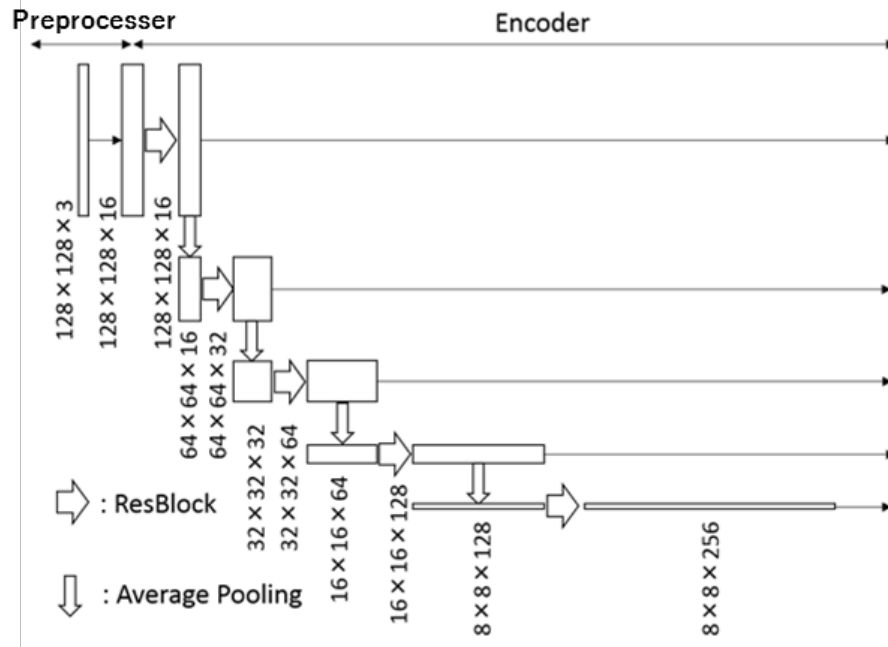


Figure 3. Architecture of our pre-processor and encoder

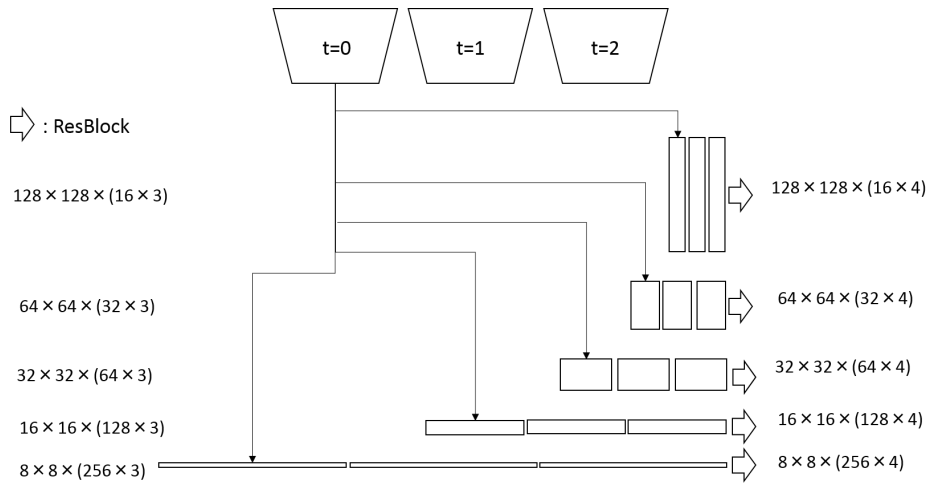


Figure 4. Architecture of our concatenator

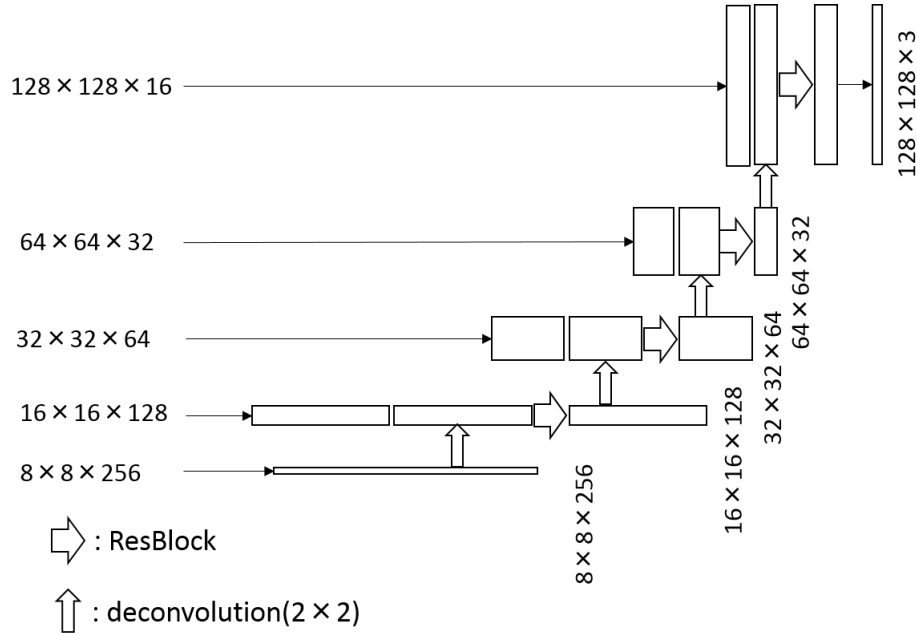


Figure 5. Architecture of our decoder

3.1.2 Discriminators

Image Discriminator: Discriminator plays the role of identifying the “ground truth” of the input images. The images produced by the generator at time steps $t=3,4,5,6$ and the real images are fed into the system.

Video Discriminator: This discriminator plays the role of identifying the “ground truth” of the input videos. The expected video at time steps $t=3,4,5,6$ and the real videos are combined and fed into the system. The Video Discriminator performs 3D convolution computations. Since four $128 \times 128 \times 3$ images are input, the input size is $128 \times 128 \times 4 \times 3$. However, for the convenience of 3D convolution, it is set to $128 \times 128 \times 128 \times 3$ by 0 padding.

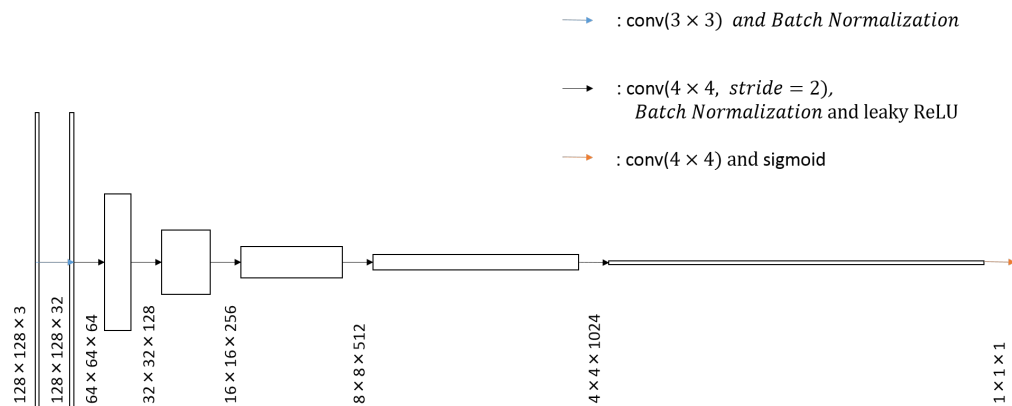


Figure 6. Architecture of Image Discriminator

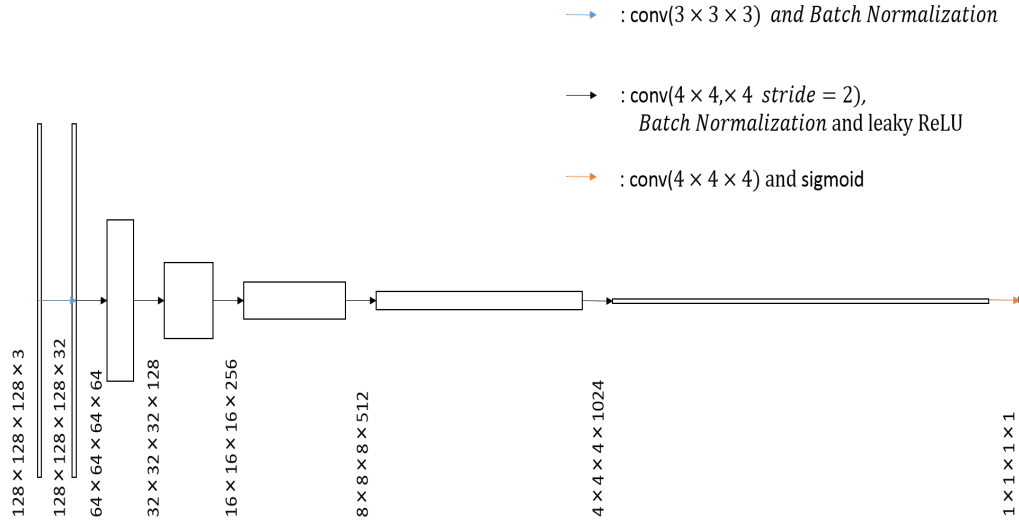


Figure 7. Architecture of Video Discriminator

3.2 Parameters Updating

The loss function of Generator is evaluated as follows:

$$L_G(G, D_i, D_v) = \lambda_1 L_{L1}(G) - \lambda_2 L_{adv}(G, D_i, D_v) \quad (4)$$

$$L_{L1}(G) = \mathbb{E}_{I, PI \sim \mathbf{I}, \mathbf{PI}}[\|I - PI\|_1] \quad (5)$$

$$L_{adv}(G, D_i, D_v) = \mathbb{E}_{(PI \sim \mathbf{PI})}[\log D_i(PI)] + \mathbb{E}_{(PV \sim \mathbf{PV})}[\log D_v(PV)] \quad (6)$$

With the L1 distance, we train Generator to bring the distance between I and PI closer. The second term of L_G encourages D_i and D_v to output 0 for a video frame from a generated image PI and a generated video PV .

We define the following loss function of Image Discriminator:

$$L_{D_i}(G, D_i) = \mathbb{E}_{(PI \sim \mathbf{PI})}[\log D_i(PI)] + \mathbb{E}_{(I \sim \mathbf{I})}[\log(1 - D_i(I))] \quad (7)$$

The first term encourages D_i to output 1 for a real image I and 0 for a generated image PI .

The loss function of Video Discriminator can be written as:

$$L_{D_v}(G, D_v) = \mathbb{E}_{(PV \sim \mathbf{PV})}[\log D_v(PV)] + \mathbb{E}_{(V \sim \mathbf{V})}[\log(1 - D_v(V))] \quad (8)$$

Similarly, the first term encourages D_v to output 1 for a real video V and 0 for a generated video PV .

We set $(\lambda_1, \lambda_2) = (100, 1)$. We use the He parameter initializer and the Adam optimizer for training. We also use the weight decay strategy. The rate of Generator and Discriminator updating is 3:1. This is to prevent the premature development of the Discriminator.

4. EXPERIMENTAL RESULTS

4.1 Dataset

In our experiment, we use LISA Traffic Light Database. The database is made for the detection of traffic light from an in-vehicle camera image. It consists of in-vehicle camera images, the coordinates of traffic lights, and information of the signal color. In our experiment, we only use in-vehicle camera images.

4.2 Result

In this experiment, the network was trained for 10 epochs (140,000 iterations). The learning result is as shown in Figure 8 and Figure 9. From the top, real images at $t = 0, 1, 2$ and the predicted images at $t = 3, 4, 5, 6$ are arranged. From Figure 8, it can be seen that the motion of the moving object can be predicted. We can also predict the movement of cars and road signs that are about to go out of the screen. Thus, we may conclude that the system can predict the motion of cars, bicycles, pedestrians, etc.



Figure 8. Prediction of moving objects (1)



Figure (8). Prediction of moving objects (2)

However, the shape of the objects such as automobile and bicycle parts appear blurred (Figure 9). Three possibilities are conceivable as the cause of the blurred output images. Firstly, there are few images of left or right turn in the training data. Of the 14,021 training data, only about 550 images deal with turning. When the training data are small, the generalization performance of the model is lowered, and the quality of prediction for the test data is lowered. This can be solved by increasing the amount of training data.

The second is the possibility that the function of the Discriminator was insufficient. Increasing the updating frequency and the ratio of output of the Discriminator in error function, and changing the structure of the Discriminator, etc. can be considered.

The third is the possibility that we expected too much work from the Generator model. In MoCoGAN, noise was generated for each moving object in the motion picture and its movement, and it was convoluted with it. MCnet handled moving objects and movements separately. In our model, we did not separate such images, and we had images generated at the same time. However, our system can predict the motion of the object, although a certain amount of blur remains regarding the shape of the object. Improvement measures can be considered by changing the configuration of the model to make it possible to recognize both movement and form.

5. CONCLUSION

As a method of predicting future camera images, we proposed a method using DCGAN. Images with $t = 0, 1, 2$ were input, and images with $t = 3, 4, 5, 6$ were predicted. With this model we were able to predict the image of the future, but although the movement of the moving object could be predicted, the shape was blurred, and it resulted in lack of accuracy. In the future it is necessary to try to build a model that can solve this problem and increase the amount of training data.

REFERENCES

- [1] S. Ren, K. He, R. Girshick, & J. Sun, (2015) Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks, NIPS.
- [2] J. Redmon, S. Divvala, R. Girshick, & A. Farhadi, (2016) You Only Look Once: Unified, Real-Time Object Detection, CVPR.
- [3] J. Redmon, & Ali Farhadi, (2018) YOLOv3: An Incremental Improvement, arXiv preprint arXiv: 1804.02767v1.
- [4] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, & S. Reed, (2016) SSD: Single Shot MultiBox Detector, ECCV.
- [5] T. Lin, P. Dollar, R. Girshick, K. He, B. Hariharan, & S. Belongie, (2017) Feature Pyramid Networks for Object Detection, CVPR.
- [6] I. J. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, & Y. Bengio, (2014) Generative Adversarial Networks, NIPS.
- [7] A. Radford, L. Metz & S. Soumith, (2015) Un-supervised Representation Learning With Deep Convolutional Generating Adversarial Networks, CoRR.
- [8] P. Isola, J. Zhu, T. Zhou, & A. A. Efros, (2017) Image-to-Image Translation with Conditional Adversarial Networks, CVPR.
- [9] O. Ronneberger, P. Fischer, & T. Brox. (2016) U-Net: Convolutional Networks for Biomedical Image Segmentation, ICLR.
- [10] S. Tulyakov, M. Liu, X. Yang, & J. Kautz, (2017) MoCoGAN: Decomposing Motion and Content for Video Generation, arXiv preprint arXiv: 1707.04993.
- [11] K. He, X. Zhang, S. Ren, & J. Sun, (2016) Deep Residual Learning for Image Recognition, CVPR.
- [12] WC. Hu, CH. Chen, TY. Chen, DY. Huang, & ZC. Wu, (2015) Moving object detection and tracking from video captured by moving camera, J. Vis. Commun. Image Represent, vol. 30, pp. 164-180.
- [13] A. Rozantsev, V. Lepetit, & P. Fua, (2014) Flying Objects Detection from a Single Moving Camera, CVPR.
- [14] S. Chen, T. Xu, D. Li, J. Zhang, & S. Jaing, (2016) Moving Object Detection Using Scanning Camera on a High-Precision Intelligent Holder, Sensors, vol. 16, no. 10, p. 1758.
- [15] R. O. Chavez-Garcia & O. Aycard, (2016) Multiple sensor fusion and classification for moving object detection and tracking, IEEE Trans. Intell. Transp. Syst., vol. 17, no. 2, pp. 525-534.
- [16] F. Fayad & V. Cherfaoui, (2008) Detection and Recognition confidences update in a multi-sensor pedestrian tracking system, Information Processing and Management of Uncertainty in Knowledge Based Systems, pp. 409-416.

- [17] R. Labayrade, D. Gruyer, C. Royere, M. Perrollaz, & D. Aubert, (2007) Obstacle Detection Based on Fusion Between Stereovision and 2D Laser Scanner, *Mobile Robots: Perception & Navigation*
- [18] S.Ioffe, & C.Szegedy, (2015) Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift, *ICML*.
- [19] D.Han, J.Kim, & J.Kim, (2017) Deep Pyramidal Residual Networks, *CVPR*.
- [20] V.Dumoulin, & F.Visin, (2016) A guide to convolution arithmetic for deep learning, arXiv preprint arXiv:1603.07285v2.

INTENTIONAL BLANK

ONLINE KNOWLEDGE-BASED EXPERT SYSTEM (KBES) FOR PSYCHOLOGICAL DISEASES DIAGNOSIS

Ahmad A. Al-Hajji¹, Fatimah M. AlSuhairani² and Nouf S. AlHarbi³

^{1,2,3}Department of Computer Science, College of Science & Arts in Al-Bukairyah,
Qassim University, Saudi Arabia

ABSTRACT

Artificial Intelligence (AI) is one of computer science branches and is used to solve problems with symbolic reasoning. Expert systems (ESs) are one of the prominent research domains of AI. We developed declarative, online procedural rule-based expert system models for psychological diseases diagnosis and classification. The constructed system exploited computer as an intelligent and deductive tool. This system diagnoses and treats more than four types of psychiatric diseases, i.e., depression, anxiety disorder, obsessive-compulsive disorder, and hysteria. The system helps psychology practitioner and doctors to diagnose the condition of a patient efficiently and in short time. It is also very useful for the patients who cannot go to a doctor because they cannot afford the cost, or they do not have a psychological clinic in their area, or they are ashamed of discussing their situation with a doctor. The system consists of program codes that make a logic decision to classify the problem of the patient. The methodology for developing the declarative model was based on the backward chaining, also called goal-driven reasoning, where knowledge is represented by a set of IF-THEN production rules. The declarative programs were written in the PROLOG. While the design of the procedural model was based on using common languages like PHP, JavaScript, CSS, and HTML. The user of the system will enter the symptoms of the patients through the user interface and the program executes. Then the program links the symptoms to the pre-programmed psychological diseases, and will classify the disease and recommend treatment.

The proposed online system link: <http://esp-online.site/>

KEYWORDS

Artificial Intelligence, Expert systems, Medical Diagnosis, Rules, Symbolic Reasoning

1. INTRODUCTION

Expert systems (ESs) belong to the class of artificial intelligence systems. ESs are complex software, which accumulate knowledge of specialists in a specific problem domain and replicating this empirical experience to consult less qualified users. Our vision is to spread the use of computer technology in the field of medical psychology to contribute to the advancement of this field. So, the proposed system is not widely implemented. Psychiatric diseases are not related to the mental and cultural development of the human being, but is mainly the result of events experienced by humans in their lives, and in many cases are painful, difficult and complex events, making him unable to solve any problem in his life, and thus exacerbate these problems and Psychiatric patients suffering from depression [1]. We tried as much as possible to explore in a deep and careful way to learn more about Psychiatric diseases to do our great research and answer questions about this disorder and will limit the content as much as possible to the following five

common types of psychiatric diseases: Depression, anxiety disorder, obsessive-compulsive disorder, and hysteria [2]. There are several factors for these Psychiatric diseases, most notably: Biological factors such as heredity, prenatal damage, infection, diseases and toxins, brain injury and congenital disabilities. Life experiences and environmental factors such as life events and emotional stress, parental abuse, and neglect, social expectations, and appreciation, poverty and societies, and cultures [3]. The purpose of this paper is create an application (Rule-Based Expert System) and to get a better understanding of exactly what expert systems do, how they are set up, and how expert system rules tend to work. The generalized structure of the expert system is presented in Figure 1. It should be noted that existing ES could have a more complex structure; however, the elements shown in this figure are an integral part of any ES. The knowledge base (KB) is the core of the ES. One of the main characteristics of this system is that in parallel to the knowledge base of the traditional expert system a database exists with information about the present state of the process that interacts with the knowledge base of the system. This database is in a state of continuous change. The knowledge base of the system contains both analytical knowledge and heuristic knowledge about the process [4].

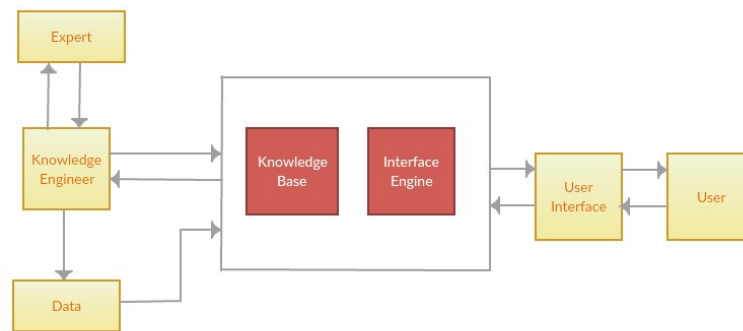


Figure 1: The generalized structure of an expert system

The knowledge engineering task comprises different knowledge sources and structures. The inference engine combines heuristic reasoning with algorithmic operation in order to reach a specific conclusion. Response time is a critical issue for online expert systems because they are operating in parallel with a dynamic process [5]. The user interface is a set of programs that implement the dialogue between the end user and the ES both at the stage of entering information and when receiving results.

2. OBJECTIVES OF THE PROPOSED SYSTEM

Following were the main objectives of the proposed expert system:

- Help the people by providing them the technology to keep in contact with their doctor or hospital, especially in the emergency;
- Design and development of an online system for diagnosing psychiatric disorders;
- Allowing the patient to recognize the symptoms of the disease he complains about;
- Use advanced software and scripting languages to make the online system more flexible and easy to interact and understand;
- Design and build a knowledge base that contains details of psychiatric diseases.

3. RELATED WORK

The medical expert systems have tolerated many changes and are using modern techniques to produce optimal outputs. It has been generally observed in various references that psychiatric diagnosis systems are based on information collected from patients.

Ahmad A, Al-Hajji [6] developed a Rule-Based Expert System for Neurological Diseases. The proposed system supports both physicians and patients with the possibility of diagnosing more than 10 types of common neurological diseases through the design of an attractive graphical interface that simulates the expert system shell.

Komal R. Hole, Vijay S. Gulhane [7] presented an Rule-based Expert System for Memory Loss Disease. This paper included the collection of information on memory loss, symptoms of memory loss and its causes by different age groups. Also, a case-based medical expert system model for the diagnosis of memory loss disease was developed. For this reason, four types of neurological diseases have been covered, i.e., Alzheimer Disease, Dementia, Parkinson Disease, Huntington Disease.

Based on the Diagnostic and Statistical Manual of Mental Disorders (DSM-IV), the references [8], [9]. [10] have developed lists of various symptoms of mental illness, that help doctors to increase the accuracy of diagnosis of these diseases, which will positively affect the condition of patients.

Dr. D.K. Sreekantha, T.M. Girish and Dr. R.V.Kulkarni [11] presented a study on knowledgebase systems in neuro science, in order to survey the soft computing techniques used in the treatment of neurological problems around the world.

Badri Adhikari, Md. Hasan Ansari, Priti Shrestha and Susma Pant [12] have developed an expert neurological diagnostic system on the Internet to help doctors diagnose nerves. Where doctors can use the website as a useful tool to diagnose their patients, using rules and cases to achieve the goal of decision-making for field experts.

Borghain, Rajdeep, and Sugata Sanyal [13] discussed the implementation of an expert-based system for the diagnosis of neurological and muscular diseases. In this system, a list of questionnaires about patients' symptoms is presented and accordingly, the disease is diagnosed and possible treatment is suggested. The system can help and support patients with neurological and muscular diseases to get an idea of their disease and possible treatment methods.

Luciano Comin Nunes, Plácido Rogério Pinheiro, Tarcísio Cavalcante Pequeno [14] presented a study aimed at making proactive decisions to adopt measures based on early diagnosis of mental disorders. Thus, it presents a proposal for some specialist system organization methodologies to support the resolution and structured representation of knowledge in the production possibilities and rules.

The book chapter [15] includes a short survey on the use of IT applications in psychology and psychiatry. Where more multidisciplinary research is being conducted in this area. The proposed system aims to increase the expert's potential by developing new methods of information finding, including application of artificial intelligence (AI) with specific HCI techniques.

4. METHODOLOGY OF THE PROPOSED SYSTEM

In this paper, it has been designed a declarative, as well as, a procedural Rule-Based Expert System (RBES) using an interactive question-and-answer sequence for diagnosis of Psychiatric diseases. Descriptive questions are used primarily to describe the existence of something or process. The questionnaire itself is based on an expert systems approach, which allows selective progress through the questionnaire structure, based on responses to previous questions. The disease symptoms have to be in a specific specialism such as Psychological Anxiety, Obsessive-Compulsive disease, Hysteria disease, and Depression [16].

An expert system for diagnosis of psychiatric diseases has been developed with two approaches: Procedural ES and Declarative ES. Design of the interfaces were written procedurally while prolog programs were written for the Declarative ES.

In this paper we present an inference engine which operates by the method of backward chaining, so the backward chaining is the best reasoning technique for diagnosis problems. Prolog is an example of a backward chaining engine. Backward Chaining is a query driven (goal-driven reasoning) approach. Beginning with a dedicated query called the goal, program rules and data items are recursively selected if they are relevant for “proving” that a query succeeds. The query is then replaced by the query part (possibly consisting of a conjunction or disjunction of smaller queries) of the selected rule, and the process is repeated until all queries can be evaluated against data items in the database (“facts”). A Prolog goal has four ports representing the flow of control through the goal (Figure 2): Call, Exit, Redo, and Fail. Prolog debuggers use these ports to describe the state of a query [17].

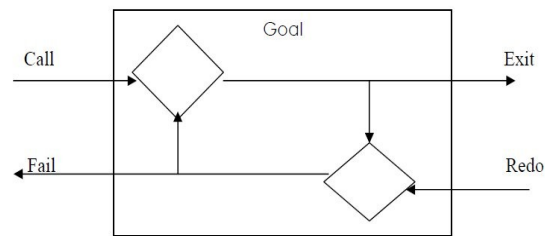


Figure 2: The ports of a Prolog goal

The explanation facility of the expert system provides a mechanism for querying the context for getting the value of a variable (what?) and knowing how a fact is established (how?) [18].

5. SYSTEM DESIGN

The proposed system has been designed in two ways: descriptive and procedural. The Figure 3. describes the components of the proposed online Rule-Based Expert System.

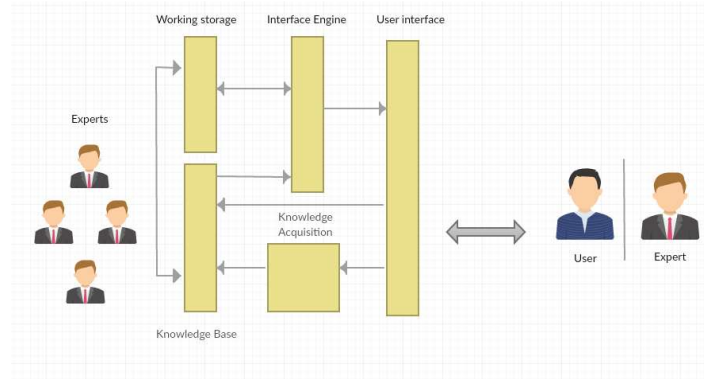


Figure 3: The proposed knowledge-based expert system structure

In this context, we can define the functions and roles of the main elements of the expert system as a framework that manages information dynamically by the integration of dedicated analysis tools. The tools to be used in any particular situation are chosen by special modules that reason about the best algorithms to use according to the information type and features. The reasoning part may be created using current Artificial Intelligence concepts and subsequently incorporated in the expert system which may also include workflows as an elementary module, as expert system techniques have matured into a standard information technology, the most important recent trend is the increasing integration of this technology with conventional information processing [19]. Here in brief we will explain the components of the expert system as follows:

The *Knowledge Base (KB)* contains the domain knowledge useful for problem solving. Each rule specifies a relation, recommendation, directive, strategy, heuristics. IF (condition) THEN (action) – When the condition part of the rule is satisfied, the rule is said to fire and the action part is executed. The *Database* includes a set of facts used to match against the IF (condition) parts of the rules stored in the knowledge base. The *Inference Engine* carries out the reasoning whereby the expert system reaches a solution. It links the rules given in the knowledge base with the facts provided in the database. The inference engine is a generic control mechanism for navigating through and manipulating knowledge and deduce results in an organized manner.

The *User Interface* controls the dialog between the user and the system. The process of *Knowledge Acquisition* is accomplished through iterations: based on feedback both from the expert and from potential users of the expert system. Performed by the knowledge engineers. The problem data stored as facts in *Working Storage* [19].

5.1 DECLARATIVE METHOD

Declarative programming often considers programs as theories of a formal logic, and computations as deductions in that logic space. Declarative programming has become of particular interest recently, as it may greatly simplify writing parallel programs [17].

Once again it should be emphasized that presented in Figure 4 structure contains only the necessary minimum, which means the mandatory presence of the elements indicated on it; If the system is presented by developers as an expert, it guarantees the availability of a knowledge processing apparatus. However, the medical ES can be significantly more difficult and additionally include databases, data exchange interfaces with various application packages, electronic libraries, etc.

5.1.1 EXPERT SYSTEM SHELL

The shell is a piece of software which contains the user interface, a format for declarative knowledge in the knowledge base, and an inference engine. The knowledge engineer uses the shell to build a system for a particular problem domain. An expert system shell (declarative) support environment should as a minimum include the following: *User Interface, Inference Engine and Knowledge Base*.

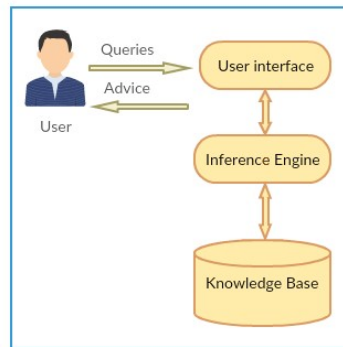


Figure 4: Declarative Expert System

5.2 PROCEDURAL METHOD

Figure 5 shows procedural online Expert System development cycle in the form of a flow chart including problem definition, conceptual design, prototyping, system development, implementation, testing and evaluating.

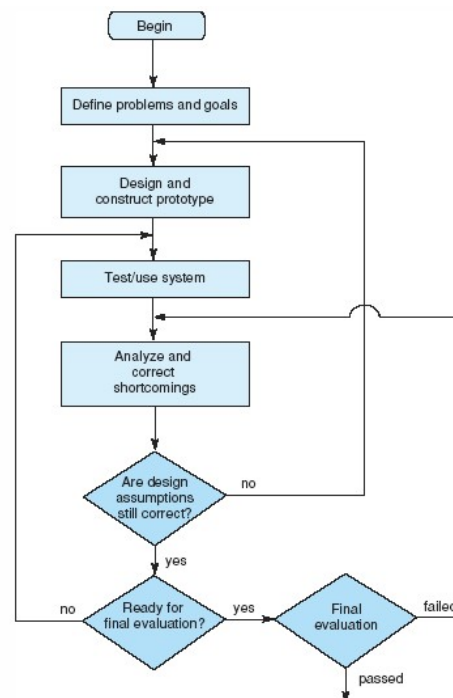


Figure 5: Flow chart of ES development cycle

5.3 PSYCHIATRIC SYMPTOMS AND DISEASES

We listed all the popular psychological symptoms (about 23 symptoms) and sorted them by their importance. The list of symptoms is illustrated in Table1. This importance was determined by the doctors at Mental Hospital Qassim (e.g. how often the symptom is observed by a patient suffering from a certain psychological disease). In the same way, we listed the diseases that depend on the previous symptoms. Some examples of Psychological Diseases are shown in Table2. After sorting, the most important combinations of the most important symptoms were formed.

Table 1: Symptoms

<i>ID</i>	<i>Symptoms</i>
1	The fatigue
2	Feeling muscle tension.
3	Hyperhidrosis
4	The confusion
5	Insomnia and difficulty of sleep
6	Abdominal pain
7	Nervousness or stress
8	Fear of being stolen or polluted
9	Fear of causing harm to others
10	Fear of mistakes
11	The excessive need for organization, integration and accuracy
12	Showers more than once
13	Hand washing frequently
14	Eating a certain fixed group of food
15	Total separation from reality
16	Temporary loss of memory
17	Absence from consciousness
18	The patient's feeling of hatred toward a certain person
19	Sleep and food disruption
20	Inactivity in the movement of the body
21	Fear and inner horror
22	Constant sense of frustration and loss of hope
23	Loss of enjoyment of life and lack of decision-making

This means that most popular clinical states would be considered. These combinations would be used as <Condition> in the rules. For each combination, the doctors then used their knowledge and experience to draw a conclusion about a patient. A program will browse the patient database to summarize the common syndromes that affirm or exclude a certain Psychological disease and then create new rules.

Table 2: Symptoms and Diseases

ID#	Disease Name	Symptoms
1	Psychological anxiety	1, 2, 3, 4, 5, 6, 7
2	Obsessive-compulsive disorder	8, 9, 10, 11, 12, 13, 14
3	Hysteria	15, 16, 17, 18
4	Depression	19, 20, 21, 22, 23

5.4 KNOWLEDGE REPRESENTATION

Decision tables are used in the elaboration of knowledge representation. A decision table for psychiatric diseases diagnosis by their symptoms (Table 3), has one row per 'rule', one column per decision variable. An additional column for the decision to take when that rule evaluates to true.

Table 3: Decision table

Diseases \ Symptoms	Feeling upset	Persistent fear	Except something to happen	Absence of consciousness	Fear of accumulation of dirt	Shut the doors continuously	Temporary loss of memory	Misalignment of limbs	Food imbalance	Frequency in making decisions	Muscle spasms	Ideas and questions	Losing hope	Increased adrenaline secretion	The feeling of hatred	Bathing ten times	Inactivity of the body	Life pressures	Fear of unknown	Hypertension	Repeat washing hands
Symbol of Symptoms	F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	F11	F12	F13	F14	F15	F16	F17	F18	F19	F20	F21
Psychological anxiety	✓	✓	✓	×	×	×	×	×	×	×	✓	×	×	✓	×	×	×	×	✓	✓	×
Obsessive-compulsive	×	×	×	×	✓	✓	×	×	×	✓	×	✓	×	×	×	✓	×	×	×	×	✓
Hysteria	×	×	×	✓	×	×	✓	✓	×	×	×	×	×	×	✓	×	×	×	×	×	×
Depression	×	×	×	×	×	×	×	×	✓	×	×	×	✓	×	✓	×	✓	✓	×	×	×

The diagnostic chart shown in Figure 6 is a graphical representation or an organization chart. The purpose of the diagnostic system is to diagnose and classify the symptoms of each disease. The diagnostic chart uses simple geometric symbols and arrows to define the relationships between each disease and its appropriate symptoms.

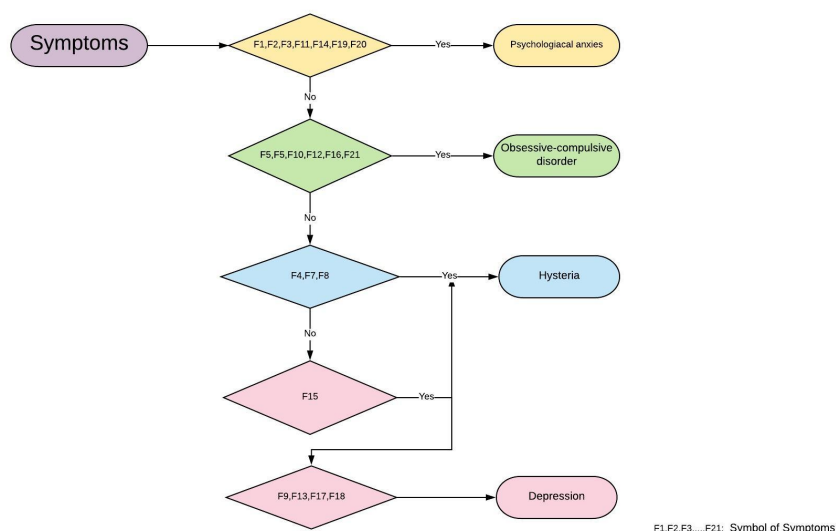


Figure 6: Diagnostic chart

5.5 REASONING UNDER UNCERTAINTY

One of the most famous expert systems is the MYCIN system, developed in the late seventies at Stanford University (USA). It is an upgraded version of the Bayesian updating, that was used in the kernel of the expert system The MYCIN system is an expert system designed to work in the field of diagnosis and treatment of blood poisoning and meningitis infections [18]. In an expert system with MYCIN certainty factors, the knowledge base consists of a set of rules that have the following syntax: *IF <evidence> THEN <hypothesis> {cf}*. Where, a cf represents belief in hypothesis H given that evidence E has occurred. In order to calculate the MYCIN certainty factor combined these weights using the formula (1) to yield a single certainty factor:

$$cf(x, y) = \left\{ \begin{array}{ll} X + Y - XY & \text{if } X > 0 \text{ and } Y > 0 \\ \frac{X + Y}{1 - \min[|X|, |Y|]} & \text{if } (X \times Y) < 0 \\ X + Y + XY & \text{if } X < 0 \text{ and } Y < 0 \end{array} \right\} \quad (1)$$

Where X and Y are the confidence in hypothesis H establish by Rule 1 & 2, respectively [20]. For example, suppose $cf(E1) = cf(E2) = 1.0$, Then, $X(H, E1) = cf(E1) * X = 1.0 \times 0.8 = 0.8$
 $Y(H, E2) = cf(E2) * Y = 1.0 \times 0.6 = 0.6$.

We obtain, $cf(x, y) = X(H, E1) + Y(H, E2) * [1 - X(H, E1)] = 0.8 + 0.6 * (1 - 0.8) = 0.92$
 Suppose the knowledge base consists of the following rules:

- Rule 1: IF A is a1
 THEN C is c1 {cf 0.8}
- Rule 2: IF B is b1
 THEN C is c1 {cf 0.6}

6. IMPLEMENTATION

The proposed KBES was designed to manage the knowledge such as creating, updating and editing the facts of users and doctors. The authors have been used different software modules, like: freely available SWI Prolog interpreter for declarative ES [21]. For online procedural ES, the codes being worked on “notepad++”, which includes several programming languages such as using PHP, MySQL, HTML, JavaScript and CSS language, were integrated to develop the procedural online ES shell model [22]. Validity of software was checked, e.g. data being acquired through various sources.

7. EXPERIMENTAL RESULTS

Two main Prolog programs were written: The first was written based on earlier types of psychological diseases and its symptoms also the relationship of the doctor with the patient (Figure 7). The second program was written for an Expert System Shell (Figure 8). Figure 9 shows SWI Prolog-Trace in action.

```

1
2 patient (name) .
3 patient (old) .
4 patient (anxietylife) .
5 patient (anxietystop) .
6 patient (uncomfortable) .
7 patient (tired) .
8 patient (impairedconcentration) .
9 patient (sleepy) .
10 patient (frequentthoughts) .
11 patient (frequentacts) .
12 patient (suffering) .
13 patient (physicalorganic) .
14 patient (depressed) .
15 patient (losttheenjoy) .
16 patient (changeappetite) .
17 patient (changesleep) .
18 patient (changeactivity) .
19 patient (fatigue) .
20 patient (lack) .
21 patient (frequentdeath) .
22
23 disease1(patient, psychologicalanxiety) .
24 disease2(patient, obsessivecompulsivedisorder) .
25 disease3(patient, hysteria) .
26 disease4(patient, depression) .
27
28 symptoms(patient1, upset) .
29 symptoms(patient1, fear) .
30 symptoms(patient1, happen) .
31 symptoms(patient1, musclespasms) .
32 symptoms(patient1, adrenalinsecretion) .
33 symptoms(patient1, fearunknown) .
34 symptoms(patient1, hypertension) .
35 symptoms(patient2, feardirt) .
36 symptoms(patient2, shutdoorscontinuously) .
37 symptoms(patient2, makingdecisions) .
38 symptoms(patient2, ideasquestions) .

```

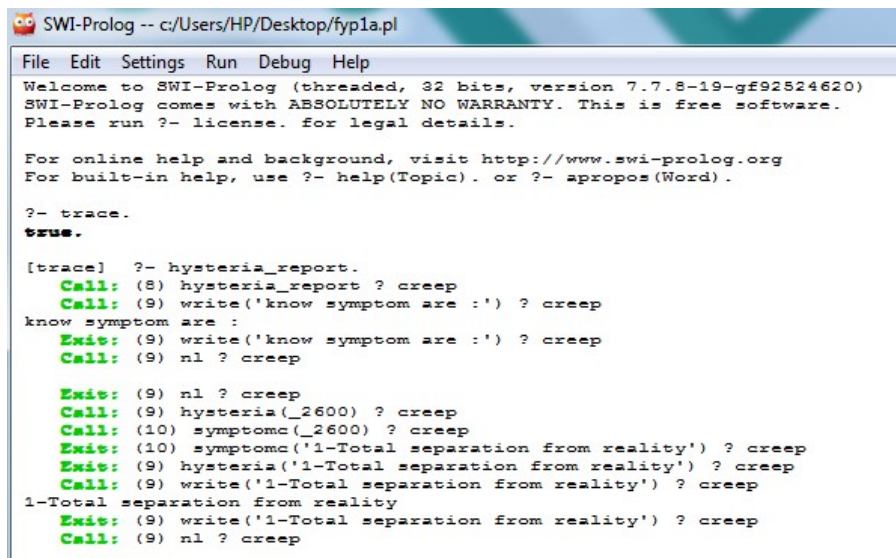
Figure 7: SWI Prolog program

```

go:-welcome,help.
welcome:-
write('    WELCOME TO PSYCHOLOGICAL EXPERT SYSTEM    '),nl.
help:-
write('What is the name of the patient?'),nl, read(NN),
write('Peace be upon you'),
write('Type your gender?'),nl, read(GG),
write('What is your age?'),nl, read(AA),
write('-----'),nl,
write('diseases available on the system are: '),nl,nl,
write('Psychological Anxiety, Obsessive Compulsive, Hysteria , Depression'),nl,
write('Diseases name_report. Show all symptoms available. '),nl, write('Symptom_no(symptom,disease name)
write('-----'),nl.
psychologicalAnxiety(X):-symptoma(X).
symptoma(X, psychologicalAnxiety).
symptoma('1-The fatigue').
symptoma('2-Feeling muscle tension').
symptoma('3-Hyperhidrosis').
symptoma('4-The confusion').
symptoma('5-Insomnia and difficulty of sleep').
symptoma('6-Abdominal pain').
symptoma('7-Nervousness or stress').
psychologicalAnxiety_report:-
write('know symptoms are :'),nl,
psychologicalAnxiety(X),
write(X),nl,fail.

```

Figure 8: Expert system shell Prolog program



```

SWI-Prolog -- c:/Users/HP/Desktop/fyp1a.pl
File Edit Settings Run Debug Help
Welcome to SWI-Prolog (threaded, 32 bits, version 7.7.8-19-gf92524620)
SWI-Prolog comes with ABSOLUTELY NO WARRANTY. This is free software.
Please run ?- license. for legal details.

For online help and background, visit http://www.swi-prolog.org
For built-in help, use ?- help(Topic). or ?- apropos(Word).

?- trace.
true.

[trace] ?- hysteria_report.
Call: (8) hysteria_report ? creep
Call: (9) write('know symptom are :') ? creep
know symptom are :
Exit: (9) write('know symptom are :') ? creep
Call: (9) nl ? creep

Exit: (9) nl ? creep
Call: (9) hysteria(_2600) ? creep
Call: (10) symptomc(_2600) ? creep
Exit: (10) symptomc('1-Total separation from reality') ? creep
Exit: (9) hysteria('1-Total separation from reality') ? creep
Call: (9) write('1-Total separation from reality') ? creep
1-Total separation from reality
Exit: (9) write('1-Total separation from reality') ? creep
Call: (9) nl ? creep

```

Figure 9: Trace in action

Following figures were some of the developed online KBES graphical user interface (GUI) snapshots. For example, Figures 10,11 show the home page and login screen of the procedural KBES respectively. When you navigate the website through the link: <http://esp-online.site/>, you can discover more interesting patterns and features for both the user and the doctor: Registration form, contact information, diagnosis and result page,...etc.

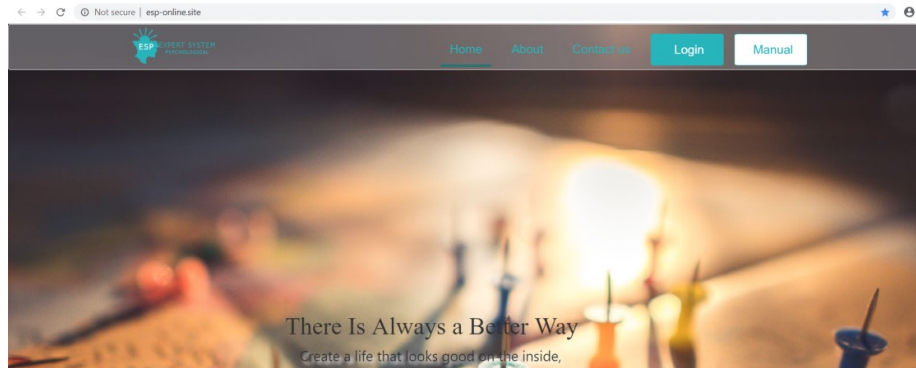


Figure 10: Home page

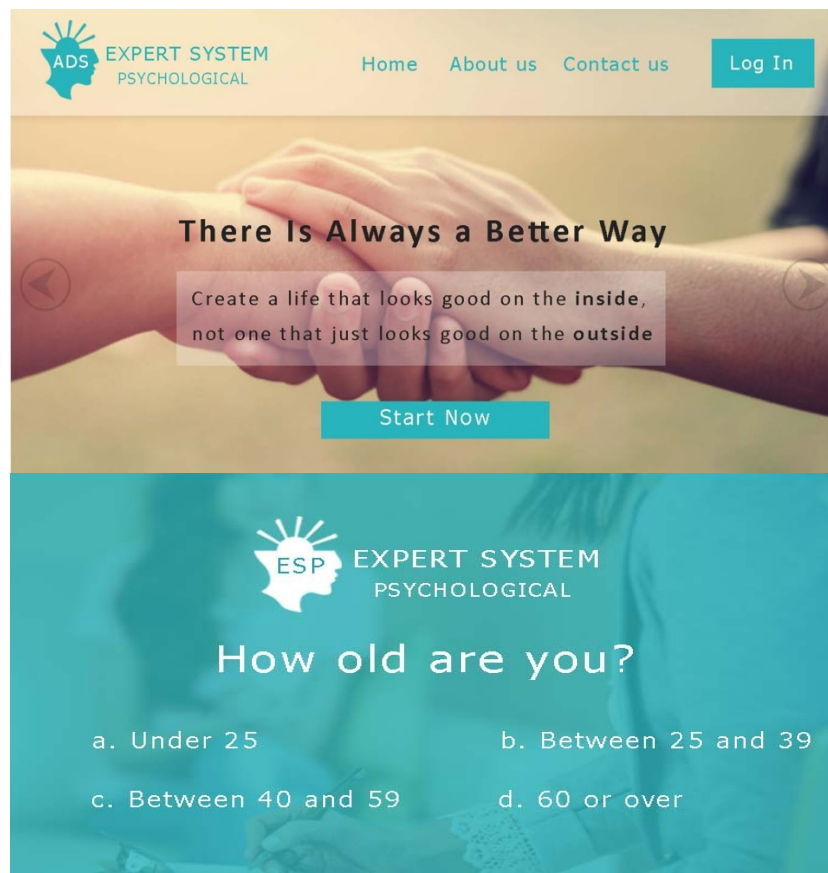


Figure 11: Login screen (patient and doctor)

The Figure 12 shows a sample diagnostic screen showing the disease name, the certainty factor, definition, where we distinguish therapy button, that takes us to a detailed report page on the diagnosed disease.



Figure 12: Result Screen

8. CONCLUSION

In this paper, declarative and an online procedural rule-based expert system models for psychological diseases diagnosis and classification were developed. The constructed system exploited computer as an intelligent and deductive tool. In this work, we have combined the benefits of Internet technology and the expert system shell. We aspire to develop and expand the services available on the Internet to psychological patients appropriately. The field of online medical experts is large and extensive, so we have a strong motivation to work on this kind of research aimed at serving psychological patients. However, the expert systems that address this type of mental illness is relatively less than others, this is one of the important points of our paper. Since the proposed expert system is a decision support tool for the right diagnosis making by the psychiatrist, the MYCIN-type uncertainty factor has been taken into account like to the Fuzzy logic. It is noted that the uncertainty factor of the declarative rule-based expert system model, appears in the form of diagnostic accuracy rate of an online procedural rule-based expert system model for different types of psychiatric diseases. The web-based expert system model can be used also as a database and archiving tool by the doctor, where the site has features beyond the psychological diseases diagnosis, can be easily detected by any observer. On future work we aim to make the diagnosis more reliable. Also, we can enhance our website to add more other types of psychiatric disorders to the knowledge base.

ACKNOWLEDGEMENTS

We would like to extend our thanks to the doctors and medical staff at Mental Hospital Qassim for their contribution and support in the mental illness information.

REFERENCES

- [1] ICD-10 Version:2016 -World Health Organization: <https://icd.who.int/browse10/2016/en#/F41.2>
- [2] Luciano CominNunes, Plácido Rogério Pinheiro, Tarcísio Cavalcante Pequeno, “An Expert System Applied to the Diagnosis of Psychological Disorders”, Conference Paper, December 2009 DOI: 10.1109/ICICISYS.2009.5358164, Source: IEEE Xplore.

- [3] Durkin, John. "Expert System: Design and Development," New York: Macmillan Publishing Company, Inc., 1994.
- [4] TMRF e-Book Advanced Knowledge Based Systems: Model, Applications & Research (Eds. Sajja & Akerkar), Vol. 1, pp 50 – 73, 2010.
- [5] Ivan Bratko, PROLOG Programming for Artificial Intelligence, 2000, ISBN 0-201-40375-7.
- [6] Al-Hajji, Ahmad A. "Rule-Based expert system for diagnosis and symptom of neurological disorders "Neurologist Expert System (NES)"." In Proceedings of the 1st Taibah University International Conference on Computing and Information Technology, Al-Madinah Al-Munawwarah, Saudi Arabia, vol. 1214, p. 6772. 2012.
- [7] Komal R. Hole, Vijay S. Gulhane, Rule-Based Expert System for the Diagnosis of Memory Loss Diseases, IJSET - International Journal of Innovative Science, Engineering & Technology, Vol. 1 Issue 3, May 2014.
- [8] Diagnostic and Statistical Manual of Mental Disorders - Fourth Edition (DSM-IV), American Psychiatric Association, Washington D.C., 1994.
- [9] International Statistical Classification of Diseases and Related Health Problems, 10th Revision, Chapter 5: Mental Disorders, Geneva, World Health Organization, 1992.
- [10] Jeste D. Press Interview. Cited by Mary Ellen Schneider in Five ways the DSM 5 could change your practice. Clin Psychiatry News. 2013:41.
- [11] Dr. D.K. Sreekantha, T.M. Girish and Dr. R.V.Kulkarni (2015), knowledgebase systems in neuro science- A study, International Journal on Soft Computing, Artificial Intelligence and Applications (IJSCAI), Vol.4, No.2, May 2015.
- [12] Badri Adhikari, Md. Hasan Ansari, Priti Shrestha and Susma Pant (September 25, 2008) "Neurology Diagnosis System", Module
- [13] Borgohain, Rajdeep, and Sugata Sanyal. "Rule based expert system for diagnosis of neuromuscular disorders." arXiv preprint arXiv: 1207.2104 (2012)
- [14] Luciano Comin Nunes, Plácido Rogério Pinheiro, Tarcisio Cavalcante Pequeno, An Expert System Applied to the Diagnosis of Psychological Disorders, DOI: 10.1109/ICICISYS.2009.5358164 • Source: IEEE Xplore
- [15] Zaharia Mihai Horia, AI Applications in Psychology, Expert Systems for Human, Materials and Automation, InTech, 2011, ISBN 978-953-307-334-7 (book chapter).
- [16] American Psychiatric Association: Diagnostic and Statistical Manual of Mental Disorders, 4th ed, revised (DSM-IV-TR). Washington, DC, American Psychiatric Association
- [17] Ivan Bratko, PROLOG Programming for Artificial Intelligence, 2000, ISBN 0-201-40375-7.
- [18] Gulankong, Dong-Ling Xu and Jian-Bo Yang, Clinical Decision Support Systems: A Review on Knowledge Representation and Inference under Uncertainties, International Journal of Computational Intelligence Systems, Vol.1, No. 2 (May, 2008), 159-167.
- [19] Michael Negnevitsky. "Artificial intelligence: A Guide to Intelligent Systems", 3rd Edition, Pearson Educational Limited, 2007.
- [20] Book: Knowledge-Based Systems, 1st Ed. Jones and Bartlett Publishers, Inc., USA ©2009 ISBN:0763776475 9780763776473.

[21] SWI-Prolog reference manual: http://www.swi-prolog.org/pldoc/doc_for?object=manual

[22] Michele E. Davis and Jon A. Phillips, Learning PHP and MySQL, Copyright © 2007, 2006 Michele E. Davis and Jon A. Phillips. All rights reserved. Printed in the United States of America., Published by O'Reilly Media, Inc., 1005 Gravenstein Highway North, Sebastopol, CA 95472

AUTHORS

Dr. Ahmad A. Al-Hajji is an Associate Professor in Computer Science Department, College of Science & Arts in Al-Bukairyah, Qassim University, Saudi Arabia. He received the B.Sc. (Eng.) degree in Electrical and Electronic Engineering from University of Aleppo 1986, his M.Sc. and Ph.D. degrees in “Automation and Computer Systems” from Odessa National Polytechnic University ONPU, Ukraine, in 1990 and 1993, respectively. He worked in different universities and countries. His research interests include Simulation and Modelling, Artificial Intelligence, ANNs, Expert Systems and Optimization Techniques.



Fatimah M. AlSuhaibani is a graduate student in the Department of Computer Science, College of Science and Arts, Al-Bukairyah. Qassim University, Saudi Arabia.

Nouf S. AlHarbi is a graduate student in the Department of Computer Science, College of Science and Arts, Al-Bukairyah. Qassim University, Saudi Arabia.

INTENTIONAL BLANK

MIXTURES OF REGRESSION CURVE MODELS FOR ARABIC CHARACTER RECOGNITION

Abdullah A. Al-Shaher

Department of Computer and Information Systems
College of Business Studies, Public Authority for
Applied Education and Training, Kuwait

ABSTRACT

In this paper, we demonstrate how regression curves can be used to recognize 2D non-rigid handwritten shapes. Each shape is represented by a set of non-overlapping uniformly distributed landmarks. The underlying models utilize 2nd order of polynomials to model shapes within a training set. To estimate the regression models, we need to extract the required coefficients which describe the variations for a set of shape class. Hence, a least square method is used to estimate such modes. We proceed then, by training these coefficients using the apparatus Expectation Maximization algorithm. Recognition is carried out by finding the least error landmarks displacement with respect to the model curves. Handwritten isolated Arabic characters are used to evaluate our approach.

KEYWORDS

Shape Recognition, Arabic Handwritten Characters, Regression Curves, Expectation Maximization Algorithm.

1. INTRODUCTION

Shape recognition has been the focus of many researchers for the past seven decades [1] and attracted many communities in the field of pattern recognition [2], artificial intelligence[3], signal processing [4], image analysis [5], and computer vision [6]. The difficulties arise when the shape under study exhibits a high degree in variation: as in handwritten characters [7], digits [8], face detection [9], and gesture authentication [10]. For a single data, shape variation is limited and cannot be captured ultimately due to the fact that single data does not provide sufficient information and knowledge about the data; therefore, multiple existence of data provides better understanding of shape analysis and manifested by mixture models [11]. Because of the existence of multivariate data under study, there is always the need to estimate the parameters which describe the data that is encapsulated within a mixture of shapes.

The literature demonstrates many statistical and structural approaches to various algorithms to model shape variations using supervised and unsupervised learning [12] algorithms. Precisely, the powerful Expectation Maximization Algorithm of Dempster [13] has widely been used for such cases. The EM Algorithm revolves around two step procedures. The expectation E step revolves around estimating the parameters of a log-likelihood function and passes it to the Maximization M

step. In a maximization (M) step, the algorithm computes parameters maximizing the expected log-likelihood found on the E step. The process is iterative one until all parameters come to unchanged. For instance, Jojic and Frey [14] have used the EM algorithm to fit mixture models to the appearances manifolds for faces. Bishop and Winn [15] have used a mixture of principal components analyzers to learn and synthesize variations in facial appearance. Vasconcelos and Lippman [16] have used the EM Algorithm to learn queries for content-based image retrieval. In general, several authors have used the EM algorithm to track multiple moving objects [17]. Revov et al. [18] have developed a generative model which can be used for handwritten character recognition. Their method employs the EM algorithm to model the distribution of sample points.

Curves are widely used in research by the computer vision society [1] [2] [3] [4] [5]. Curvatures are mainly used to distinguish different shapes such as characters [6], digits, faces [2], and topographic maps [3]. Curve fitting [18][19] is the process of constructing a 2nd order or higher mathematical function that best fits to a series of landmark points. A related topic is a regression analysis which stresses on probabilistic conclusion on how uncertainty occurs when fitting a curve to a set of data landmarks with marginal errors. Regression curves are applied in data visualization [12][13] to capture the values of a function with missing data [14] and to gain relationship between multiple variables.

In this paper, we demonstrate how curves are used to recognize 2D handwritten shapes by applying 2nd order of polynomial quadratic function to a set of landmark points presented in a shape. We then train such curves to capture the optimal characteristics of multiple shapes in the training set. Handwritten Arabic characters are used and tested in this investigation.

2. REGRESSION CURVES

We would like to extract the best fit modes that describe the shapes under study, hence, multiple image shapes are required and explained through training sets of class shape ω and complete sets of shape classes denoted by Ω . Let us assume that each training set is represented by the following 2D training patterns as a long vector

$$X^\omega = ((x_1^{\omega_1}, y_1^{\omega_1}), \dots, (x_k^{\omega_1}, y_k^{\omega_1}), (x_1^{\omega_2}, y_1^{\omega_2}), \dots, (x_k^{\omega_2}, y_k^{\omega_2}), (x_1^{\omega_T}, y_1^{\omega_T}), \dots, (x_k^{\omega_T}, y_k^{\omega_T})) \quad (1)$$

Our model here is a polynomial of a higher order. In this example, we choose 2nd order of quadratic curves. Consider the following generic form for polynomial of order j

$$f(x_k^{\omega_T}) = a_0 + a_1 x_k^{\omega_T} + a_2 (x_k^{\omega_T})^2 + a_3 (x_k^{\omega_T})^3 + \dots + a_j (x_k^{\omega_T})^j = a_0 + \sum_{\tau=1}^j a_\tau (x_k^{\omega_T})^\tau \quad (2)$$

The nonlinear regression above requires the estimation of the coefficients that best fit the sample shape landmarks, we approach the least square error between the data y and $f(x)$ in

$$err = \sum (d_i)^2 = (y_1^{\omega_T} - f(x_1^{\omega_1}))^2 + (y_2^{\omega_T} - f(x_2^{\omega_1}))^2 + (y_3^{\omega_T} - f(x_3^{\omega_1}))^2 + (y_4^{\omega_T} - f(x_4^{\omega_1}))^2 \quad (3)$$

where the goal is to minimize the error, we substitute the form of equation (3) with a general least square error

$$err = \sum_{k=1}^T (y_k^{\omega_T} - (a_0 + a_1 x_k^{\omega_T} + a_2 x_k^{\omega_T^2} + a_3 x_k^{\omega_T^3} + \dots + a_j x_k^{\omega_T^j}))^2 \quad (4)$$

where T is the number of pattern set, k is the current data landmark point being summed, j is the order of polynomial equation. Rewriting equation (4) in a more readable format

$$err = \sum_{k=1}^T (y_k^{\omega_T} - (a_0 + \sum_{k=1}^j a_k x^k))^2 \quad (5)$$

Finding the best fitting curve is equivalent to minimize the squared distance between the curve and landmark points. The aim here is to find the coefficients, hence, solving the equations by taking the partial derivative with respect each coefficient a_0, \dots, a_k ; for $k = 1 \dots j$ and set each to zero in

$$\frac{\partial err}{\partial a_0} = \sum_{k=1}^T (y_k^{\omega_T} - (a_0 + \sum_{K=1}^T a_K x^k)) = 0 \quad (6)$$

$$\frac{\partial err}{\partial a_1} = \sum_{k=1}^T (y_k^{\omega_T} - (a_0 + \sum_{K=1}^T a_K x^k)) x_k^{\omega_T} = 0 \quad (7)$$

$$\frac{\partial err}{\partial a_2} = \sum_{k=1}^T (y_k^{\omega_T} - (a_0 + \sum_{K=1}^T a_K x^k)) x_k^{\omega_T^2} = 0 \quad (8)$$

Rewriting upper equations in the form of a matrix and applying linear algebra matrix differentiation, we get

$$\begin{bmatrix} T & \sum_{k=1}^T x_k^{\omega_T} & \sum_{k=1}^T x_k^{\omega_T^2} \\ \sum_{k=1}^T x_k^{\omega_T} & \sum_{k=1}^T x_k^{\omega_T^2} & \sum_{k=1}^T x_k^{\omega_T^3} \\ \sum_{k=1}^T x_k^{\omega_T^2} & \sum_{k=1}^T x_k^{\omega_T^3} & \sum_{k=1}^T x_k^{\omega_T^4} \end{bmatrix} \begin{bmatrix} a_0 \\ a_1 \\ a_2 \end{bmatrix} = \begin{bmatrix} \sum_{k=1}^T y_k^{\omega_T} \\ \sum_{k=1}^T x_k^{\omega_T} y_k^{\omega_T} \\ \sum_{k=1}^T x_k^{\omega_T^2} y_k^{\omega_T} \end{bmatrix} \quad (9)$$

Choosing Gaussian elimination procedure to rewrite the upper equation in more solvable in

$$Ax = B \quad (10)$$

where

$$A = \begin{bmatrix} T & \sum_{k=1}^T x_k^{\omega_T} & \sum_{k=1}^T x_k^{\omega_T^2} \\ \sum_{k=1}^T x_k^{\omega_T} & \sum_{k=1}^T x_k^{\omega_T^2} & \sum_{k=1}^T x_k^{\omega_T^3} \\ \sum_{k=1}^T x_k^{\omega_T^2} & \sum_{k=1}^T x_k^{\omega_T^3} & \sum_{k=1}^T x_k^{\omega_T^4} \end{bmatrix}, X = \begin{bmatrix} a_0 \\ a_1 \\ a_2 \end{bmatrix}, B = \begin{bmatrix} \sum_{k=1}^T y_k^{\omega_T} \\ \sum_{k=1}^T x_k^{\omega_T} y_k^{\omega_T} \\ \sum_{k=1}^T x_k^{\omega_T^2} y_k^{\omega_T} \end{bmatrix} \quad (11)$$

solving X to find the coefficients A, B in

$$X = A^{-1} * B \quad (12)$$

The outcome would be the coefficients a_0, a_1, a_2 . We follow the similar procedure to find the coefficient sets of the remaining landmark points.

3. LEARNING REGRESSION CURVES

It has been acknowledge that when using learning algorithms to train models of such case, the outcome is trained models with superior performance to those of untrained models Bishop [19]. In this stage, we are concerned with capturing the optimal curve coefficients which describe the patterns variations under testing; hence, training is required, thereby, fitting the Gaussian mixtures to curve coefficient models to a set of shape curve patterns. The previous approaches consider producing variations in shapes in a linear fashion. To obtain more complex shape variations, we have to proceed by employing non-linear deformation to a set of curve coefficients. Unsupervised learning is encapsulated in a framework of the apparatus Expectation Maximization EM Algorithm. The idea is borrowed from Cootes [20] who was the pioneer in constructing point distribution models; however, the algorithm introduced by Cootes[20] is transformed to learn regression curves coefficients α_t similar to that approach of AlShaher [21]. Suppose that a set of curve coefficients α_t for a set of training patterns is $t = (1 \dots T)$ where T is the complete set of training curves is represented in a long vector of coefficients :

$$\alpha_t = (a_{t_{11}}, a_{t_{12}}, a_{t_{13}}, a_{t_{21}}, a_{t_{22}}, a_{t_{23}}, \dots, a_{t_{in}}, a_{t_{in}}) \quad (13)$$

The mean vector of coefficient patterns is represented by

$$\mu = \frac{1}{T} \sum_{t=1}^T \alpha_t \quad (14)$$

The covariance matrix is then constructed by

$$\sum = \frac{1}{T} \sum_{t=1}^T (\alpha_t - \mu) (\alpha_t - \mu)^T \quad (15)$$

The following approach is based on fitting a Gaussian mixture models to a set of training examples of curve coefficients. We further assume that training patterns are independent from one to another; thus, they are neither flagged nor labelled to any curve class. Each curve class ω belongs to a set of curve classes Ω has its own mean μ and covariance matrix \sum . With these component elements. For each curve class, we establish the likelihood function for a set of the curve patterns in

$$p(\alpha_t) = \prod_{t=1}^T \sum_{w=1}^{\Omega} p(\alpha_t | \mu_w, \Sigma_w) \quad (16)$$

Where the term $p(\alpha_t | \mu_w, \Sigma_w)$ is the probability of drawing curve pattern α_t from the curve-class ω . Associating the above likelihood function with the Expectation Maximization Algorithm, the likelihood function can be a process reformed in two steps. The process revolves around estimating the expected log-likelihood function iteratively in

$$q_L(C^{(n+1)} | C^{(n)}) = \sum_{t=1}^T \sum_{w=1}^{\Omega} P(\alpha_t, \mu_w^{(n)}, \Sigma_w^{(n)}) X \ln p(\alpha_t | \mu_w^{(n+1)}, \Sigma_w^{(n+1)}) \quad (17)$$

Where the quantity and $\mu_w^{(n)}$ and $\Sigma_w^{(n)}$ are the estimated mean curve vector and variance covariance matrix both at iteration (n) of the Algorithm. The quantity $p(\alpha_t, \mu_w^{(n)}, \Sigma_w^{(n)})$ is the *a posteriori* probability that the training pattern curve belongs to the curve-class ω at iteration n of the Algorithm. The term $p(\alpha_t | \mu_w^{(n+1)}, \Sigma_w^{(n+1)})$ is the probability of distribution of curve-pattern α_t belongs to curve-class ω at iteration (n + 1) of the algorithm; thus, the probability density is associated with the curve-patterns α_t for (t = 1 ... T) to curve-class ω are estimated by the updated construction of the mean-vector $\mu_w^{(n+1)}$, and covariance matrix $\Sigma_w^{(n+1)}$ at iteration n+1 of the algorithm. According to the EM algorithm, the expected log-likelihood function is implemented in a two iterative processes. In the M or maximization step of the algorithm, our aim is to maximize the curve mean-vector $\mu_w^{(n+1)}$, and covariance matrix $\Sigma_w^{(n+1)}$, while, in the E or expectation step, the aim is to estimate the distribution of curve-patterns at iteration n along with the mixing proportion parameters for curve-class ω .

In the E, or Expectation step of the algorithm, the a posteriori curve-class probability is updated by applying the Bayes factorization rule to the curve-class distribution density at iteration n+1. The new estimate is computed by

$$p\left(\alpha_t, \mu_w^{(n)}, \sum_w^{(n)}\right) = \frac{p(\alpha_t | \mu_w^{(n)}, \Sigma_w^{(n)}) \pi_w^{(n)}}{\sum_{w=1}^{\Omega} p(\alpha_t | \mu_w^{(n)}, \Sigma_w^{(n)}) \pi_w^{(n)}} \quad (18)$$

where the revised curve-class ω mixing proportions $\pi_w^{(n+1)}$ at iteration (n + 1) is computed by

$$\pi_w^{(n+1)} = \frac{1}{T} \sum_{t=1}^T p(\alpha_t | \mu_w^{(n)}, \Sigma_w^{(n)}) \quad (19)$$

With that at hand, the distributed curve-pattern α_t to the class-curve ω is Gaussian distribution and is classified according to

$$p\left(\alpha_t | \mu_w^{(n)}, \sum_w^{(n)}\right) = \frac{1}{(2\pi)^L \sqrt{|\Sigma_w^{(n)}|}} \exp \left[-\frac{1}{2} (\alpha_t - \mu_w^{(n)})^T X \left(\sum_w^{(n)} \right)^{-1} X (\alpha_t - \mu_w^{(n)}) \right] \quad (20)$$

In the M , or Maximization step, our goal is to maximize the curve-class ω parameters. The updated curve mean-vector $\mu_w^{(n+1)}$ estimate is computed using the following

$$\mu_w^{(n+1)} = \sum_{t=1}^T p(\alpha_t, \mu_w^{(n)}, \Sigma_w^{(n)}) \alpha_t \quad (21)$$

And the new estimate of the curve-class covariance matrix is weighted by

$$\Sigma_w^{(n+1)} = \sum_{t=1}^T p(\alpha_t, \mu_w^{(n)}, \Sigma_w^{(n)}) X(\alpha_t - \mu_w^{(n)}) (\alpha_t - \mu_w^{(n)})^T \quad (22)$$

Both E, and M steps are iteratively converged, the outcome of the learning stage is a set of curve-class ω parameters such as $\mu_w^{(n)}$ and $\Sigma_w^{(n)}$, hence the complete set of all curve-class Ω are computed and ready to be used for recognition.

With the stroke and shape point distribution models to hand, our recognition method proceeds in a hierarchical manner.

4. RECOGNITION

In this stage, we focus on utilizing the parameters extracted from the learning phase to obtain in shape recognition. Here, we assume that the testing shapes

$$f(t) = \sum_{t=1}^X \sum_{i=1}^n (x_{t_i}, y_{t_i}), \text{ where } (i = 1 \dots n), (t = 1 \dots X) \quad (23)$$

Hence, each testing pattern is represented by

$$\chi_t = ((x_{t_1}, y_{t_1}), (x_{t_2}, y_{t_2}), \dots (x_{t_i}, y_{t_i})) \text{ for } (t = 1 \dots X) \quad (24)$$

Such testing patterns are classified according to computing the new point position of the testing data χ after projecting the sequence of curve-coefficients by

$$f(x, y) = \sum_{i=1}^n (\chi_{t_{y_i}} - (\alpha_{t_{i1}} \chi_{t_{x_i}}^2 + \alpha_{t_{i2}} \chi_{t_{x_i}} + \alpha_{t_{i3}})) \quad (25)$$

So the sample shape χ_i is registered to class ω which has the highest probability using Bayes rule over the total curve-classes Ω in

$$\arg \min_{\omega \in \Omega} \frac{f(x, y)}{\sum_{w=1}^{\Omega} f(x, y)} \quad (26)$$

5. EXPERIMENTS

We have evaluated our approach with sets of Arabic handwritten characters. Here, we have used 23 shape-classes for different writers, each with 80 training patterns. In total, we have examined the approach with 1840 handwritten Arabic character shape patterns for training and 4600 patterns for recognition phase. Figures 1 illustrates some training patterns used in this paper. Figure 2 demonstrates single shapes and their landmarks representation.

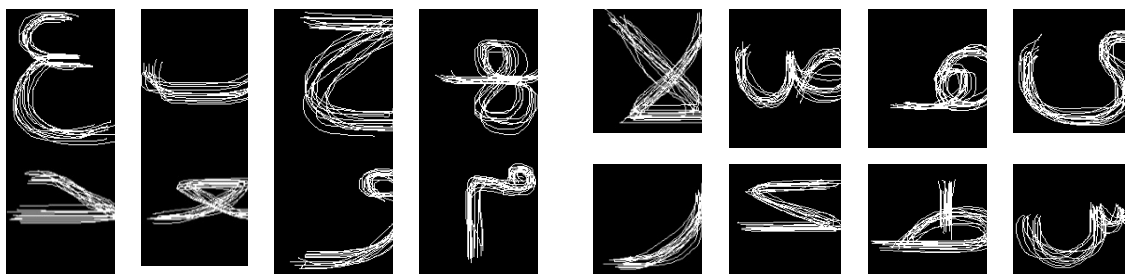


Figure 1: Training sets sample

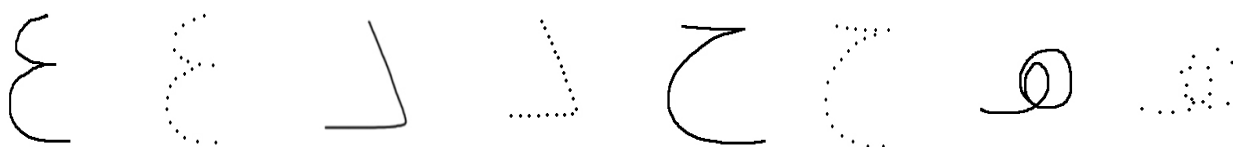


Figure 2: Training patterns and extracted landmarks

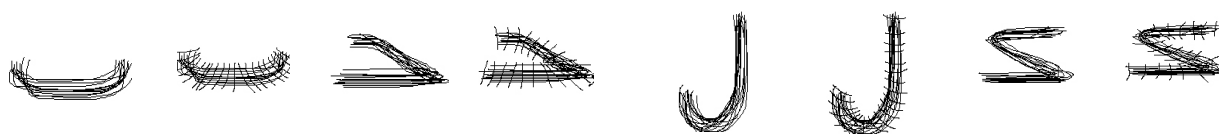


Figure 3: Sample visual of regression Curve-classes

Figure 3 indicates regression sample curve-classes as a result of the training stage. Figure 4 demonstrates the curve-classes Ω convergence rate graph as a function per iteration no. in the training phase. The graphs shows how associated distributed probabilities for the set of curve-classes Ω converged into a few iterations.

To take this investigation further, we demonstrate how well the approach behaves in the presence of noise. In figure 5, we show how recognition rate is achieved when point position displacement error is applied. Test shape coordinates are being moved away from their original position. The figure proves that the recognition rate fails to register shapes to their correct classes in a few iterations and it decreases completely when coordinates are moved away, yet, increasing variance significantly.

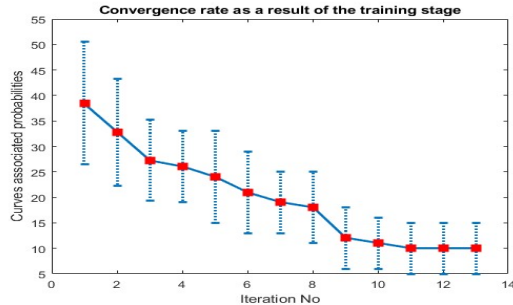


Figure 4: Convergence Rate as a function per iteration no.

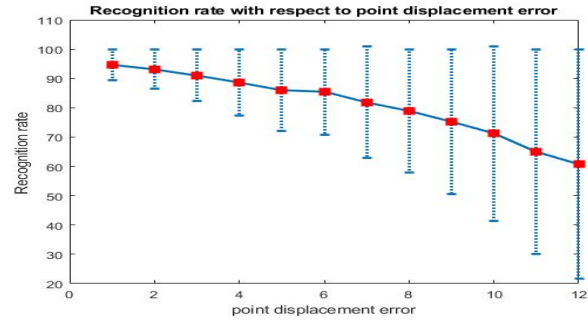


Figure 5: Recognition rate as a function per iteration no with point position error.

Table 1 shows recognition rates per curve-classes ω . Table 1 demonstrates recognition rates per curve-class. In total, we have achieved 94% recognition rate with this approach.

Table 1: Recognition Rates for sample shapes

Sample Shape	Test Size	Correct	False	Recognition Rate	Sample Shape	Test Size	Correct	False	Recognition Rate
	200	191	9	95.5%		200	176	24	88%
	200	193	7	96.5%		200	175	25	87.5%
	200	183	17	91.5%		200	172	28	86%
	200	187	13	93.5%		200	181	19	90.5%
	200	196	4	98%		200	190	10	95%
	200	180	20	90%		200	182	18	91%
	200	178	22	89%		200	193	7	96.5%

6. CONCLUSION

In this paper, we have proved how Regression Curves can be utilized to model the variation of Handwritten Arabic characters. A 2nd order of Polynomials curves are injected along the skeleton of the proposed shape under study, where the appropriate set of curve-coefficients which describe the shape were extracted. We, then have used the Apparatus of the Expectation Maximization Algorithm to train the set of extracted set of curve-coefficients within a probabilistic framework to capture the optimal shape variations coefficients. The set of best fitted parameters are then projected to recognize handwritten shapes using Bayes rule of factorization. The proposed approach has been evaluated on sets of Handwritten Arabic Shapes for multiple different writers by

which we have achieved a recognition rate of nearly 94% on corrected registered shape classes.

ACKNOWLEDGMENT

I would like to express my sincere gratitude to the Public Authority for Applied Education and Training (Research Department) for their full funding to this research. Their kind support including hardware, software, books, and conference fees allowed me to investigate and conduct this research with significant results. Their attention permitted me to contribute a good result to the literature of Pattern Recognition field specifically in recognizing handwritten Arabic characters.

REFERENCES

- [1] J. Wood, "Invariant pattern recognition: A review," *Pattern Recognition*, vol. 29, no. 1, 1996, 1-17.
- [2] Anil, K. Jain, Robert P.W. Duin, and Jianchang Mao: "Statistical Pattern Recognition: A Review", *IEEE Pattern Analysis and Machine Intelligence*, vol 22, No. 1, PP 4-37, 2000.
- [3] P.F. Baldi and K. Hornik, "Learning in linear neural networks: A survey," *IEEE Transactions on Neural Networks*, vol. 6, no. 4, 1995, 837-858.
- [4] T.Y. Kong and A. Rosenfeld, "Digital topology: introduction and survey," *Computer Vision, Graphics, and Image Processing*, vol. 48, no. 3, pp. 357-393, 1989.
- [5] T.R. Reed and J.M.H. Dubuf, "A review of recent texture segmentation and feature extraction techniques," *CVGIP - Image Understanding*, vol. 57, no. 3, pp. 359-372, 1993.
- [6] S. Sarkar and K.L. Boyer, "Perceptual organization in computer vision - a review and a proposal for a classifactory structure", *IEEE Transactions on Systems Man and Cybernetics*, vol. 23, no. 2, 1993, 382-399.
- [7] Lorigo L. M., Govindaraju V., "Offline Arabic Handwriting Recognition: A Survey", *IEEE Trans Pattern Anal Mach Intelligence*, Vol: 28(5): PP 712-24, 2006.
- [8] Ishani Patel, Virag Jagtap, Ompriya Kale, "A Survey on Feature Extraction Methods for Handwritten Digits Recognition", *International Journal of Computer Applications*, Vol107, No12, PP 11-17, 2014.
- [9] Muhammad Sharif, Farah Naz, Mussarat Yasmin, Muhammad AlyasShahid, AmjadRehman, "Face Recognition: A Survey", *Journal of Engineering Science and Technology Review*, Vol 10, No 2, PP 166-177, 2017.
- [10] Rafiqul Zaman Khan, Noor Adnan Ibraheem, "HAND GESTURE RECOGNITION: A LITERATURE REVIEW", *International Journal of Artificial Intelligence & Applications*, Vol.3, No.4, pp 161-174, 2012.
- [11] Bruce George Lindsay, Mary L. Lesperance, "A review of semiparametric mixture models", *Journal of Statistical Planning and Inference*, Vol: 47, No: 1 PP 29-39, 1995.
- [12] Christopher M. Bishop, "Neural Networks for Pattern Recognition", Clarendon Press, 1995, ISSN 0198538642.
- [13] A. Dempster, N. Laird, D. Rubin, Maximum likelihood from incomplete data via the em algorithm, *J. Roy. Statist. Soc. Ser. 39* (1977) 1-38
- [14] J. Brendan, N. Jovic, Estimating mixture models of images and inferring spatial transformations using the em algorithm, *IEEE Comput. Vision Pattern Recognition 2* (1999) 416-422.

- [15] C. Bishop, J. Winn, Non-linear Bayesian image modelling, Proceedings of Sixth European
- [16] N. Vasconcelos, A. Lippman, A probabilistic architecture for content-based image retrieval, Proceedings of International Conference on Computer Vision and Pattern Recognition, 2000, pp. 216–221.
- [17] B. North, A. Blake, Using expectation-maximisation to learn dynamical models from visual data, Image Vision Computing. 17 (8) (1999) 611–616.
- [18] M. Revow, C. Williams, G.E. Hinton, Using generative models for handwritten digit recognition, IEEE Trans. Pattern Anal. Mach. Intell. 20 (2) (1996) 592–606
- [19] Christopher M. Bishop, Pattern Recognition and Machine Learning, Springer Science, and Business Media, 2006.
- [20] T. Cootes, C. Taylor, A mixture model for representing shape variations. Image and Vision Computing, 17(1999) 403-409
- [21] AlShaher Abdullah, Hancock Edwin, Learning mixtures of Point Distribution models with the EM algorithm. Pattern Recognition, 36(2003) 2805-2818.

DETECTION OF HATE SPEECH IN SOCIAL NETWORKS: A SURVEY ON MULTILINGUAL CORPUS

Areej Al-Hassan¹ and Hmood Al-Dossari²

¹Department of Information Systems, King Saud University, Riyadh, Saudi Arabia

²Department of Information Systems, King Saud University, Riyadh, Saudi Arabia

ABSTRACT

In social media platforms, hate speech can be a reason of “cyber conflict” which can affect social life in both of individual-level and country-level. Hateful and antagonistic content propagated via social networks has the potential to cause harm and suffering on an individual basis and lead to social tension and disorder beyond cyber space. However, social networks cannot control all the content that users post. For this reason, there is a demand for automatic detection of hate speech. This demand particularly raises when the content is written in complex languages (e.g. Arabic). Arabic text is known with its challenges, complexity and scarcity of its resources. This paper will present a background on hate speech and its related detection approaches. In addition, the recent contributions on hate speech and its related anti-social behaviour topics will be reviewed. Finally, challenges and recommendations for the Arabic hate speech detection problem will be presented.

KEYWORDS

Text Mining, Social Networks, Hate Speech, Natural Language Processing, Arabic NLP

1. INTRODUCTION

Over the last decades, people are getting more engaged with the wide spread of social networks. Microblogging applications opened up the chance for people around the globe to express and share their thoughts extensively and in a real-time manner. Such expressions afford researchers with the ability to investigate the online social emotions in different events. People now have the potential to speak freely, this allowed them to exchange all sorts of thoughts, emotions and knowledge. However, cyberspace is not always safe, it can be a reason for the dissemination of aggressive and harmful content. Hate speech is an online common form for expressing prejudice and aggression. This may convey racist, xenophobic and many forms of verbal aggression. Hate speech is typically defined as the act that disparages a person or people on the basis of a number of characteristics that may include and not limited to: race, ethnicity, sexual orientation, gender, religion and nationality [1]. In social media platforms, there are uncontrollable number of comments and posts issued every second which make it impossible to trace or control the content of such platform. Therefore, social platforms are facing a problem in limiting these posts while balancing the freedom of speech[2]. In addition, the diversity of people and their backgrounds, cultures and beliefs can ignite the flame of hate speech [3]. In the other hand, each culture has its own different interpretations and characteristics of cyber-hate. So, every culture is assumed to act differently and have their own way of intervention in a manner which best suits the culture.

For the Arab region, there is a noticeable growth in the usage of social media platforms. According to the Arab social media report [4], the social media penetration in the Arab region reached 90% of the population in some countries. This increase in usage and the openness in speech results in a public concern on existing practices in social networks. A vast amount of posts that is hard and even impossible to control manually by platform owners. Hate and aggression through social networks should be rationed and regulated by policy makers and should be also countered by harnessing the power of artificial intelligence and machine learning algorithms to automate the detection of hate speech in social media.

The rest of the paper is organized as follows: section 2 will include a theoretical background, section 3 will go through the works related to hate speech detection. Then a discussion and recommendations will be presented in section 4. Finally, we conclude this paper by highlighting the future research directions.

2. BACKGROUND

2.1. What is Hate Speech

The case of hate speech and violent communication conducted over the internet can be referred as cyber-hate [5]. It is a narrow and specific form of cyber-bullying and it can be defined as “any use of electronic communications technology to spread racist, religious, extremist or terrorist messages” it is different from cyber-bullying in that hate speech can target not only individuals but it also has implications on whole communities [1]. Brown [6] has also defined hate speech as any textual or verbal practice that implicates issues of discrimination or violence against people in regard to their race, ethnicity, nationality, religion, sexual orientation and gender identity. According to Anis [7] hate speech can occur in different linguistic styles and several acts like insulting, provocation, abusing and aggression. However, according to Chetty and Alathur [8], hate speech can be categorized into the following categories:

2.1.1. Gendered hate speech

This category includes Any form of hostility towards particular gender or any devaluation based on person's gender. This include any post that offense particular gender. Also it includes any form of misogyny. Moreover, Jha and Mamidi [9] clarify that sexism may come in two forms: Hostile (which is an explicit negative attitude) and Benevolent (which is more subtle).

2.1.2. Religious hate speech

This will include any kind of religious discrimination, such as: Islamic sects, calling for atheism, Anti-Christian and their respective denominations or anti-Hinduism and other religions. However, Albadi et al. [10] mentioned that religious hate speech is considered as a motive of crimes in countries with highest social crimes.

2.1.3. Racist hate speech

Lastly, this category includes is Any sort of racial offense or tribalism, regionalism, xenophobia (especially for migrant workers) and nativism (hostility against immigrants and refugees) and any prejudice against particular tribe or region. For instance, offending an individual because he belongs to a particular tribe or region or country or favoritism of a particular tribe. Add to that, offending the appearance and color of individual.

2.2. What Constitutes Hate Speech

Hate speech is hard to comprehend. However, it can be recognized based on specific characteristics that can be distinguished from one culture to another. These characteristics are debatable, some may interpret them as a pure hate and some don't. This problem is considered as a controversial problem that no one can agree upon. Gelashvili and Nowak [11] argued that it is an obstacle for social media platforms owners to regulate hate speech as many questions will raise to their heads such as what constitute hate speech? And what kind of hate speech need to be countered? Only legitimate people who are actively engaged in the same culture and who can be competent enough can give the answers to these questions. Some studies have given some necessary terminologies for studying hate speech, for example Fortuna and Nunes [12] have listed some of the main rules for hate speech identification. In brief, hate speech is identified when disparaging stereotype about group. Together with using racial and sexist slurs with intent to harm. Add to that when indecently speak about religion or specific country.

Correspondingly, when identifying hate speech, we need to exclude some conditions. For instance, when trying to explain the meaning of some abusive words or when we use some of racial terms in another context which has no hate undertone. Add to that when writing a news article and referring to a sect which is associated with hate crime "e.g. ISIS" this referral itself won't be considered as hate speech. In like manner, Waseem and Hovy [2] have proposed 11 parameters to distinguish hate speech specifically in twitter platform, some of which are: usage of sexist and racial terms, attacking and criticizing minority, promoting violence, distorting the truth with lies and supporting suspicious hashtags.

Given these characteristics, a reasonable list can be derived for a particular culture with certain adjustments to deal with the controversy and then from that list, hate speech can be reliably identified and recognized. Anis [7] discussed the dominant themes in Arabic hate speech particularly in the newspaper and concluded that hate speech in Arab region is generally related to religion and sectarian themes.

2.3. Text Mining and NLP for Hate Speech Detection

The problem of hate speech in social networks is technically considered as unstructured text problem. Therefore, extracting insights and pattern from such text can be a bit challenging, owing to the context-dependent interpretation of natural language. Text mining technologies have the capabilities to handle the ambiguity and variability of unstructured data [13].

Natural Language Processing or (NLP) is the main pillar of text mining, it employs a number of computational tasks in order to make human natural language tractable and understood by the machine [14]. Today, NLP researchers have moved towards the rich and controversial data available in social networks by downloading vast amount of unstructured data, these data can be mined and put into practical use. Text mining for social networks requires a number of lexical, syntactic and semantic NLP tasks aiming to give a structure to the text for further processing. These tasks include: Tokenization, which splits the text into word tokens by the spaces. Also, in this task, stop words like "in", "the" will be taken out as they make no sense to the meaning. There are a number of available tokenization tools such as Apache "OpenNLP¹" or "Stanford Tokenizer²". Then, predicting Part of Speech (PoS) for each token will take a place in aim to provide lexical information. Then, parsing will take a place by representing the syntactic structure of the whole text [15]. A significant drawback of NLP nowadays is that most of the tools are

¹ <https://opennlp.apache.org>

² <https://nlp.stanford.edu/software/tokenizer.shtml>

exclusively designed for common languages such as English, French, Spanish [14]. Comparatively, uncommon language such as Arabic has a challenge associated with the difficulty in adapting the common languages tools. However, Arabic linguistics experts have gone through a considerably good achievements in analyzing Arabic language morphology [16]. In particular, Khoja stemmer [17] and a stemmer by Ghwanmeh et al. [18] and finally, AlKhalil Morpho system [19] which is considered as the best Arabic Morphological system [16].

2.4. Arabic Text in Social Networks

Arabic language has a very high growth rate in means of usage in social networks. Based on the Arab Social Media Report [4] the average rate of using Arabic language in social media reaches 55% in 2017. As it can be seen, the amount of Arabic content in social networks is growing substantially in recent years. Facebook stands as the most popular platform in the Arab region, followed by Twitter, LinkedIn and Instagram, with penetration rates (34%, 13%, 6.75%,1.8%) respectively.

Arabic language is known by its difficulties and challenges. In case of twitter, Salem [4] stated that it is hard to extract meaningful insights from an Arabic tweet, basically because tweets are very noisy and people don't care about spelling and grammar in their posts. Secondly, they contain great amount of variances including: writing from right to left, combining Arabic with Latin words and the usage or the neglection of diacritics [20]. Not to mention the different local informal dialect for each Arab country, this issue can be considered as the major issue for Arabic language, especially when we are considering hate speech, some Arabic terms may imply hateful meaning in one region, while it is considered an ordinary term in others. Consequently, Salem [4] claimed that many researchers tend to work with specific Arab region and try to fine-tune the used algorithms to adapt this specific region aiming to increase the accuracy of their works.

2.5. Automatic Hate Speech Detection in Social Networks

One of the main applications of social media mining is the automatic detection of events and behaviors which includes identifying people behavior in real-world events through monitoring their interactions with each other. Researchers can take an advantage of these explosive data to reach substantial insights [21]. This task depends mainly on text mining approaches such as NLP and machine learning algorithms. In twitter, researchers explored many automatic detection tasks, such as: anti-social behaviour detection, spam detection, natural disasters (e.g. earthquakes), trends and public opinion events. To achieve this task, several features and common patterns need to be identified. Then, machine learning algorithms are applied to perform the classification task to get the targeted result out of the data.

2.5.1. Features representation for hate speech detection

To perform an automatic detection task such as hate speech detection general features of the corpus need to be specified in order to enable the classification algorithms to perform the task. Some of these approaches will be presented.

Dictionaries and Lexicons. This feature usually employed in unsupervised machine learning scenarios [22]. Wiegand et al. [23] addressed the detection of profane words by taking advantage of the corpora and lexical resources. They used several features and general-purpose lexical resource to build their lexicon. Usually lexicon-based approaches are not competitive to other features used in supervised approaches since they are domain independent. Gitari et al. [24] also used a lexicon as a primary feature by aggregating opinions and giving rates to the subjective words.

Bag-of-words (BOW) and N-grams. It can be considered as word co-occurrence feature. A vectorization process is performed on tokenized words in the corpus by assigning weight for each word according to its frequency in the tweet and its frequency in between different tweets, the vectorization process is done using some statistical models (e.g. TF-IDF weight). After that, a list of words together is called BOW which will be presented as vectors of weights [25]. N-gram representation means a sequences of N adjacent words. Waseem and Hovy [2] analyzed the impact of using number of features in conjunction with character N-gram for detecting hate speech. They found that using character n-gram representation is a great option for detecting hate speech. BOW is limited by its need to be accompanied with other features to improve the performance, but in the other hand it is computationally expensive [26]. For N -grams, it needs careful selection for the value of N to avoid high level of distance between related words [27].

Latent Dirichlet Allocation (LDA). It is a probabilistic topic modeling method. It is mainly used to give an estimation of the latent topics in data set and these latent topics will be used as features instead of words. However, LDA is suitable for unsupervised and semi-supervised machine learning settings. Xiang et al. [28] claimed that BOW did not work well for abusive text detection in twitter. Instead, they include highly expressive topical feature and other lexicon features by using LDA model. this approach can be an alternative for that supervised methods.

Word embedding and Word2Vec. The emergence of word embedding mitigated the data sparsity problem by bringing up an extra semantic feature by generating distributed representations that introduces dependence between words. Word2Vec is one of the techniques to construct word embedding. According to Lilleberg et al. [29] word2vec has given a lot of interest by researchers in text mining field and it is compatible with both supervised and unsupervised machine learning models.

2.5.2. Machine learning for hate speech detection

After preparing the text to work with machine, classification algorithms can take a place to perform the detection task. In terms of classifiers, machine learning approaches can be categorized into: supervised, semi-supervised and unsupervised approaches.

Supervised learning. This approach is domain dependent since it relies on a manual labeling of a large volume of text. Labeling task is time and effort consuming but it is more efficient for domain-dependent events. Most of the approaches used for hate speech detection tasks are supervised methods. For instance, Burnap and Williams [30] have used several supervised classifiers to detect hate speech in twitter, their results showed that all classifiers have performed the same but the different settings of features changed the accuracy of the model. Consequently, the choice of the classifier depends on the features that can be extracted from the corpus.

Semi-supervised learning. In this paradigm, algorithms are trained using both of labeled and unlabeled data. Using labeled data in conjunction with unlabeled data can effectively enhance the performance, this can be seen in Hua et al. [31] model. They argued that unsupervised learning has limited ability to handle small scale events. On the contrary, supervised learning has the capability to effectively capture small scale events but the need to manually label the data set decreases the scalability of the model. To achieve the right balance between these two situations authors suggested a semi-supervised approach. Moreover, Xiang et al. [28] replaced the costly manual annotation with an automatically generated feature, They claimed that their approach can be a good alternative to the costly supervised approaches to detect hate speech.

Unsupervised learning. It is a domain-independent approach and is capable to handle a diversity of content while maintaining scalability [32]. It does not rely on human labor to label a large volume training set, instead, it dynamically extracts domain-related key terms. Gitari et al. [24] utilized a bootstrapping approach to build their lexicon by starting with small seed of hate verb and then expanded it iteratively. The best results from their model were obtained when they incorporated semantic hate and them-based features.

2.5.3. Deep learning

Deep learning models show promising future in text mining tasks. It depends entirely on the artificial neural networks but with extra depth. It tries to mimic the event in layers of neurons and attempt to learn in a real sense to identify patterns in the provided text. However, deep learning approaches are not always better than the traditional supervised approaches. The performance of deep learning is subject to the right choice of algorithm and number of hidden layers as well as the feature representation technique. Al-Smadi et al. [33] proved the previous assumption by comparing the performance of both Recurrent Neural Network (RNN) and Support Vector Machine (SVM). Their comparison showed that SVM outperformed RNN for specific set of features. So, they suggested to use (LSTM) and different algorithm for the embedding for their future work. For hate speech detection, Pitsilis et al. [34] used RNN model with word frequency vectorization to implement the features instead of the word embedding to break the barrier of language dependency in word embedding approach. Their results outperformed the current state of art deep learning approaches for hate speech detection.

3. RELATED WORK

This section presents a comprehensive review on the key works and existing studies related to the area of automatic detection and hate speech in particular.

3.1. Current state in Hate Speech Detection and Related Concepts

There are some researches that have discussed different related terminologies which serves similar related concept to the phenomena of hate speech (e.g. cyber-bullying, abusive language, radicalization detection). The analysis of these different terminologies will definitely help to reach insights from different perspectives in current situation and will also contribute in spotting and recognizing the interrelationship among these terminologies.

3.1.1. Abusive language detection

It is the general concept that covers all the hurtful language. Hate speech is considered under the umbrella of abusive language. This terminology also covers profanity (the use of inappropriate words). However, many researches refer to abusive language as offensive language. Chen et al. [35] used YouTube comments as a dataset to detect offensive language. They used a combination of lexical and syntactic features and they incorporated user's writing style to predict user's behaviour in the future. Also, Wiegand et al. assumed that they can filter abusive words from the negative polar expressions. They took advantage of a base lexicon by taking a small subset of negative polar expressions and then via crowdsourcing, the abusive words were labelled. Similar approach was proposed by Xiang et al. [28] to detect offensive content in twitter. Their features were mainly based on the linguistic regularities of the profane terms also based on statistical topic modelling on a relatively big dataset. For a deep learning scenario, Park and Fung [36] compared the performance of one-step and two-step classifiers by using the dataset provided by Waseem and Hovy [2]. Based on their results, they believe that combining 2 classifiers (e.g. CNN and logistic regression) can boost up the performance. Moreover, Chen et al. [37] used FastText as

their neural network classifier to detect abusive text from various social networks platforms. They found that FastText performance is lower than using SVM as a classifier.

3.1.2. Cyberbullying detection

The electronic form of traditional bullying is called cyberbullying, which is the aggression and harassment that is targeted to an individual who is unable to defend himself [38]. Bullying is known with its repetitive act to the same individual, unlike hate speech which is more general and not necessarily intended to hurt a specific individual. Dinakar et al. [39] research is one of the pioneers and most cited researches for the textual cyberbullying detection. Their experiment was based on a corpus of 4500 YouTube comments. Their result showed that showing the polarities of the dataset outperformed categorizing the dataset into a multiclass. Both of Nahar et al. [40] and Capua et al. [41] presented unsupervised approach for cyberbullying detection. Özel et al. [42] work is unique to this area because they have investigated Turkish language in order to detect cyberbullying from twitter and Instagram text. Their results showed that Naïve Bayes Multinomial showed the best results in both accuracy and total training and testing time. Finally, Pawar et al. [43] utilized distributed computing for cyberbullying detection. Their work focuses mostly on the robust performance rather than the accuracy alone.

3.1.3. Radicalization detection

This concept is usually referred to as a motive towards violent extremism. Usually radical groups have an ideology that considers violence as a legitimate action when it serves to address their concerns [44]. Radicalization and hate speech are closely related and usually mentioned as if they have the same meaning but actually radicalization comes under hate speech as it has specific tendencies towards religious beliefs. Wadhwa and Bhatia [45] referred to radical groups as “cyber-extremists”. They investigated the possibility of the detection of such act in Twitter using unsupervised approach. They came with the conclusion that fully unsupervised approach will not be able to detect the right topics for this issue, manual intervention is necessary to reach better results because tweets have a dynamic nature. Agarwal and Ashish [46] introduced a semi-supervised approach to detect radicalization in twitter. They had a mixture of labeled and unlabeled data. They mainly counted on the hashtags with radical tendencies (e.g. #Terrorism) to identify extremism promoting tweets. Fernandez and Alani [47] believe that the major reason behind the inaccuracy of previous approaches that detect radicalization is because they mainly rely on the appearance of terminologies and expressions regardless of their context.

3.1.4. Hate speech detection

Starting from the early stages, Warner and Hirschberg [48] were one of the first initiatives to automate the detection of hate speech in the World Wide Web. Their research was specifically to detect anti-Semitic and their work included a number of challenges that should be overcome by now. The first racial oriented research was by Kwok and Wang [49] who decided to follow Warner and Hirschberg path and implemented a supervised model to detect racist tweets. Then, Burnap and Williams [50] were motivated to investigate the spread of hate speech immediately after Lee Rigby murder in UK. They trained a supervised classifier to find hateful and non-hateful tweets related to this particular event. Waseem and Hovy’s work [2] was a baseline research for many following researchers as they have investigated the predictive features for hate speech detection. They allowed the access to their huge corpus of 16K tweets that is dedicated for hate speech researches in English language.

Hate speech in other languages was also investigated. Del Vigna et al. [51] designing a model to detect hate speech Italian language in Facebook. Their result showed that the classifiers

were not able to discriminate between three levels of hate. In addition, Alfina et al. [52] investigated the ability to detect hate in Indonesian language. They took advantage of a political event to collect their sample “Jakarta Governor election 2017”. For the German hate speech Jaki [53] initiated a quick response to the recent German NetDG law by proposing a model to detect hate speech in German language. He also established a comprehensive qualitative and quantitative analysis in what constitute hate speech from the political communication perspective.

Recent works have shifted to the employment of deep learning for such task, Gambäck and Sikdar [54] applied deep learning approach on Waseem and Hovy’s dataset [2]. Their results outperformed by means of precision and recall. The same corpus was also used by Badjatiya et al. [55] for comparing different combinations of deep learning models. In addition, Zhang et al. [56] explored combining convolutional and gated recurrent unit networks, they also compared their model performance with all previous deep learning models. They stated that their work sets a new benchmark for future researches in this area. Finally, Pitsilis et al. [34] believe that deep neural networks have a high potential to solve the issue of hate speech detection. Their deep learning approach outperformed all the state-of-art approaches.

3.2. Arabic Hate Speech Detection

A limited number of Arabic researches have contributed to that particular area. In the other hand, many Arabic researches were investigated in similar areas which we can call “Anti-social behaviors” such as, Abusive or offensive language and cyberbullying.

3.2.1. Arabic anti-social behaviour detection

Starting with Abusive language detection. Abozinadah paved the way in this area and contributed in three researches tailored for this area. First, Abozinadah et al. [57] proposed a model in response to Arab governments needs of blocking such abusive contents. They created their own test set and made it publicly available. Then in [58] Abozinadah and H. Jones, Jr. enhanced the previous work by proposing a lexicon that is fed by an Arabic word correction method to enhance the detection of such abusive words. A third work by Abozinadah is [59] which used statistical learning approach for the detection process to overcome the limitation in the BOW approach presented in other previous works. Mubarak et al. [60] work aimed to build a large scale corpus of Arabic tweets that are classified to (Obscene, offensive and clean) and made it available for next researchers.

Another two contributions by Alakrot et al. [61][62]. In the first work, they have constructed a corpus of Arabic comments from YouTube and made it publicly available for abusive detection purposes. In their second work, an empirical examination of the dataset has been performed. They concluded that a combining N-gram and stemming may results in lower performance. Alshehri et al. [63] followed the same path of Abozinadah and Azalden but the concept is slightly different. They created a large scale of adult content in Arabic Twitter. Consequently, a large lexicon was built based on that corpus.

In addition to the previous behaviors, cyberbullying is another serious issue that has been addressed by many researchers in English language. For the Arabic language, Haidar et al. [64] made the first attempt to detect cyberbullying in Arabic language. Their work was the first step into this area, since it needs a lot of enhancements such as considering more features related to cyberbullying and choosing better feature representation. Alduailej and Khan [65] discussed the main challenges of detecting cyberbullying in Arabic language. The fundamental challenge was that we need to discover the context before deciding whether it is considered cyberbullying or not.

Finally, radicalization and extremism are another two anti-social behaviors that have been studied in Arabic language scenarios. Magdy et al. [66] classified twitter users whether they are supporting or opposing ISIS by discriminating the language that shows support for ISIS. Kaati et al. [67] proposed a model that detects whether a user is more likely to support Jihadist groups. Their experiment showed that AdaBoost classifier worked well for English tweets but it did not give the expected performance in Arabic tweets.

3.2.2. Arabic hate speech detection

In English language, hate speech detection has been intensively investigated by more than 14 contributors who investigated all the categories of hate speech (racial, sexism, religious and general hate). In contrast, Arabic language has limited available resources for detecting various categories of hate speech, actually, only one contribution has been found in this area which is specifically targeted to the detection of “Religious” Arabic hate speech. Albadi et al. [10] were the first to tackle the problem of religious hatred in Arabic twitter, but they didn’t encounter the other categorizations of hate speech. They built and scored a lexicon of the most common religious terms. They tested various classifiers for this task including GRU RNN which outperformed the rest of classifiers. They stated their reasons behind choosing GRU rather than LSTM, they claimed that GRU works better with smaller datasets and it is faster with respect to training time, also it has lower probability to overfit small datasets.

3.3. Summery and Analysis

The next tables present a summary of all the discussed papers and they are organized according to their respective time series. These tables cover the following topics respectively: English Anti-social behaviours, English hate speech and finally, Arabic Anti-social behaviours. These tables can serve as a quick reference for all the key works done in the automatic detection in social media. All the approaches and their respective experiments results are listed in a concise manner. Table 1 consolidates all the terminologies related to hate speech and their corresponding contributions. Table 2 summarizes all the multilingual contributions and papers which are directly related to hate speech. Finally, table 3 which gives an emphasize on the Arabic language by summarizing all the works that deals with the detection of anti-social behaviour in social media platforms. For the results column, the best results in each paper is pointed.

Table 1. Summary of the current state of anti-social behaviour detection, and their respective results, in the metric: Precision (P), Recall (R), F1-Score (F).

Author	Year	Platform	ML approach	Features Representation	Algorithm	P	R	F
Abusive Language (English)								
Chen et al. [35]	2012	YouTube	Un-Supervised	Lexical and syntactic	Match Rules	0.98	0.94	-
Xiang et al.[28]	2012	Twitter	Semi-Supervised	Topic modelling	Logistic Regression	-	-	0.84
Park, Fung [36]	2017	Twitter	Supervised	Character and Word2vec	Hybrid CNN	0.71	0.75	0.73
Chen et al. [37]	2017	Youtube, Myspace, SlashDot	Supervised	Word embeddings	FastText	-	0.76	-
Wiegand et al.[23]	2018	Twitter, Wikipedia, UseNet	Supervised	Lexical, linguistics and word embedding	SVM	0.82	0.80	0.81

Cyberbullying (English)								
Dinakar et al.[39]	2011	YouTube	Supervised	Tf-idf, lexicon, PoS tag, bigram	SVM	0.66	-	-
Nahar et al. [40]	2014	Myspace, Slashdot	Semi-Supervised	Linguistic features	Fuzzy SVM	0.69	0.82	0.44
Capua et al.[41]	2016	YouTube, Form-Spring, Twitter	Un-Supervised	Semantic and syntactic features	GHSOM network and K-mean	0.60	.094	0.74
Pawar et al.[43]	2018	Form-spring	Supervised	Bag of words	M-NB and Stochastic Gradient Descent	-	-	0.90
Cyberbullying (Turkish)								
Özel et al.[42]	2017	Twitter, Instagram	Supervised	Bag of words	M-Naïve Bayes	-	-	0.79
Radicalization (English)								
Wadhwa , Bhatia [45]	2013	Twitter	Un-Supervised	Topic identification, N-grams	Topic-entity mapping	-	-	-
Agarwal , Sureka [46]	2015	Twitter	Semi-Supervised	Linguistic, Term Frequency	LibSVM	-	-	0.83
Fernandez and Alani [47]	2018	Twitter	Supervised	Semantic Context	SVM	0.85	0.84	0.85

Table 2. Summary of the current state of hate speech and their respective results, in metrics: Precision (P), Recall (R), F1-Score (F).

Author	Year-Platform	Classes	ML Approach	Features Representation	Algorithm	P	R	F
Religious hate speech (English)								
Warner and Hirschberg [48]	2013-Yahoo news-group	Anti-Semitic, not anti-Semitic.	Supervised	Template-based, PoS tagging	SVM	0.59	0.68	0.63
Racial hate speech (English)								
Kwok and Wang[49]	2013-Twitter	Racist, Non-racist	Supervised	Unigram	Naïve Bayes	-	-	-
General hate speech (English)								
Burnap and Williams [50]	2014-Twitter	Yes, No	Supervised	BOW, Dependencies, Hateful Terms	Bayesian Logistic Regression	0.89	0.69	0.77
Gitari et al. [24]	2015-Blog	No hate, Weakly hate, Strongly hate	Semi-Supervised	Lexicon, Semantic, theme-based features	Rule based	0.73	0.68	0.70

Djuric et al. [68]	2015-Yahoo Finance	Hateful, Clean	Supervised	Paragraph2vec, CBOW	Logistic regression	-	-	-
Waseem and Hovy[2]	2016-Twitter	Hate, not hate	Supervised	Character n-grams	Logistic regression	0.72	0.77	0.73
Watanabe et al.[3]	2018-Twitter	Hateful, Offensive, Clean	Supervised	Sentiment-Based, Semantic, Unigram,	J48graft	0.79	0.78	0.78
Malmasi and Zampieri [69]	2018-Twitter	Hate, offensive, Ok	Supervised	N-grams, Skip-grams, hierarchical word clusters	RBF kernel SVM	0.78	0.80	0.79
Gambäck and Sikdar [54]	2017-Twitter	Non-hate, Racism, Sexism, Both	Supervised	Character N-grams, word2vec	CNN	0.85	0.72	0.78
Badjatiya et al. [55]	2017-Twitter	Sexist, Racist, Neither sexist nor racist	Supervised	Random Embedding,	LSTM and GBDT	0.93	0.93	0.93
Pitsilis et al. [34]	2018-Twitter	Neutral, Racism or Sexism	Supervised	Word-based frequency vectorization	RNN and LSTM	0.90	0.87	0.88
Zhang et al. [70]	2018-Twitter	Racism, Sexism, Both, Non-hate	Supervised	Word embeddings	CNN+GRU	-	-	0.94
General hate speech (Italian)								
Del Vigna et al. [51]	2017-Facebook	Hate, Not hate	Supervised	Morpho-syntactical, sentiment polarity, word embedding lexicons.	SVM	0.75	0.68	0.71
					RNN and LSTM	0.70	0.75	0.72
General hate speech (Indonesian)								
Alfina et al. [52]	2017-Twitter	Hate speech, Non-hate speech	Supervised	BOW and n-gram	Random Forest Decision Tree	-	-	0.93
General hate speech (German)								
Jaki. [53]	2018-Twitter	Muslim, Terrorist, Islamofascistoid	Un-Supervised	Skip grams and Character trigrams	K-means, single-layer averaged Perceptron	0.84	0.83	0.84

Table 3. Summary of the Arabic contributions in anti-social behaviour detection and their respective results, in the metrics: Precision (P), Recall (R), F1-Score (F).

Author	Year-Platform	Classes	ML Approach	Features Representation	Algorithm	P	R	F
Abusive language (Arabic)								
Abo-zinadah et al. [57]	2015-Twitter	Abuser, Normal	Supervised	Profile and tweet-based features, bag of words, N-gram, TF-IDF	Naïve Bayes	0.85	0.85	0.85
Abo-zinadah and H. Jones, Jr. [58]	2016-Twitter	Abusive, Legitimate Accounts	Un-Supervised	Lexicon, bag of words (BOW), N-gram	SVM	0.96	0.96	0.96
Abo-zinadah and H. Jones, Jr.[59]	2017-Twitter	Non-Abusive, Abusive	Supervised	PageRank (PR) algorithm, Semantic Orientation (SO) algorithm, statistical measures.	SVM	0.96	0.96	0.96
Mubarak et al.[60]	2017-Twitter, Arabic News Site	Obscene, Offensive and Clean	Un-supervised	unigram and bigram, Log Odds Ratio (LOR), Seed Words lists	None. Just performed extrinsic evaluation	0.98	0.45	0.60
Alakrot et al. [62][61]	2018-YouTube	Offensive, In-offensive	Supervised	N-gram	SVM	0.88	0.80	0.82
Violent content (Arabic)								
Abdelfatah et al. [71]	2017-Twitter	Violent, Non-violent	Un-supervised	Sparse Gaussian process latent variable model, morphological features, Vector Space Model	K-means clustering	0.56	0.60	0.58
Adult content (Arabic)								
Alshehri et al.[63]	2018-Twitter	Adult, Regular user	Supervised	Lexicon, N-grams, bag-of-means (BOM)	SVM	0.70	0.93	0.78
Cyberbullying (Arabic)								
Haidar et al.[64]	2017-Facebook, Twitter	Yes, No	Supervised	Tweet to SentiStrength Feature Vector	SVM	0.93	0.94	0.92
Terrorism (Arabic)								
Magdy et al. [66]	2016-Twitter	Pro-ISIS and Anti-ISIS	Supervised	Temporal patterns, Hashtags	SVM	0.87	0.87	0.87
Kaati et al. [67]	2016-Twitter	Support or Oppose Jihadism	Semi-Supervised	Data dependent features and data independent features.	AdaBoost	0.56	0.86	0.86

Religious hate speech (Arabic)								
Albadi et al.[10]	2018-Twitter	Hate, Not hate	Supervised	Word embeddings (AraVec)	GRU-based RNN	0.76	0.78	0.77

4. DISCUSSION AND FUTURE WORK

After exploring the literature, Arabic hate speech detection challenges can be pointed out based on what have been discussed in previous works.

4.1. Arabic Hate Speech Detection Challenges

Hate speech detection is not a simple keyword spotting, it is a complex task with many challenges. Based on the review conducted in the previous section, we can spot several research challenges in the automated detection of Arabic hate in social media.

First barrier is that there are a few of researches in hate speech detection that can result in high precision and recall rates. These few researches are mostly dedicated for languages with Latin characters, on the other hand, there is a gap in the Arabic language researches in this area. Many researchers confessed that Arabic language is the major challenge due to its complexity and richness in both of its derivations and inflections, add to that, the varieties of dialects used by Arab users in twitter. Secondly, Arabic hate speech detection is a multidisciplinary problem and it needs to be investigated from different dimensions, one of the challenges is related to the social and political perspective, will we be able to discriminate different hate speech contexts for different Arab cultures? As there is no unified definition for what constitutes hate speech. Coupled with the issue of legitimacy, hate speech contains a broad and loose range of expressions and sometimes, trivial issues can be included and considered as part of it which makes it hard to discriminate which case is more critical.

Moving to the technical perspective, choosing the most appropriate machine learning approach is another challenging decision. Previous works employed mostly all the varieties of techniques. According to tables 1,2,3, majority of researchers relied on supervised machine learning approaches in their automatic detection task. Un-supervised approaches come to the next place of popularity and semi-supervised approaches are the least used techniques. We need to consider all the factors that can affect our decision in the right choice of the approach. For instance, one major factor is the size of the corpus, as some ML algorithms works pretty well with small datasets. Others such as Neural Networks needs more intensive and complex training.

Recent researches are oriented towards deep learning to solve complex learning tasks. Researchers claimed that deep learning is powerful when it comes to finding data representation for classification and obviously it has a promising future in the field of the automatic detection. Choosing to adopt deep learning needs commitment in both of preparing and training the model with large amount of data. Generally, there are two main architectures for deep neural networks that are usually utilized for NLP tasks, these models are: RNN and CNN. In the previous tables, there were 4 hate speech researches that adopted deep learning, two of them were RNN and the two others were CNN. These researches concluded with the effectiveness of both approaches. for that reason, more investigation needs to be done to make the appropriate choice of deep learning architecture.

4.2 Machine Learning Model

When working with a specific language (e.g. Arabic) and particular region, this task can be considered as domain-dependent task. Consequently, supervised approaches are the best candidates for this task. However, Since the revolution of deep learning and deep neural networks for NLP tasks, we will consider narrowing our choice to these robust models. Yin et al. [72] conducted a comparative study between the two deep neural networks “RNN and CNN” as they are the most commonly used deep learning models for NLP tasks. Basically, RNN has two types: GRU and LSTM and it supports sequential architectures. CNN in the other hand has a hierarchical architecture. Yin experiments showed that RNN are well suited for the long-ranged context dependencies. While CNN is better in extracting local features. GRU and CNN results can be compared with respect to text size, GRU is better when the sentences are bit longer. Finally, they concluded that deep neural network performance is highly dependable on tuning the hyperparameters.

4.3 Conclusion and Future Work

Arab regions and worldwide are now more aware of the problem of spreading hate through the social networks. Many countries are working hard in regulating and countering such speech. This attention raised the need for automating the detection of hate speech. In this paper we analyzed the concept of hate speech and specifically “cyber hate” which is conducted in the means of social media and the internet sphere. Moreover, we differentiated between the different anti-social behaviors which include (Cyberbullying, Abusive and offensive language, Radicalization and hate speech). After that we presented a comprehensive study on how text mining can be used in social networks. we investigated some challenges which can be a guide for the implementation of Arabic hate speech detection model. In addition, these recommendations will help in drawing a road map and a blueprint for the future model. The future work will include incorporating the latest deep learning architectures to build a model that is capable to detect and classify Arabic hate speech in twitter into distinct classes. A data set will be collected from twitter, and for intensifying the training of our neural network we will including data from additional platform “e.g. Facebook” as it is the most used platform in the Arab region.

ACKNOWLEDGEMENT

The authors would like to thank Deanship of scientific research in King Saud University, for funding and supporting this research through the initiative of DSR Graduate Students Research Support (GSR).

REFERENCES

- [1] C. Blaya, “Cyberhate: A review and content analysis of intervention strategies,” *Aggress. Violent Behav.*, no. May, pp. 0–1, 2018.
- [2] Z. Waseem and D. Hovy, “Hateful Symbols or Hateful People? Predictive Features for Hate Speech Detection on Twitter,” *Proc. NAACL Student Res. Work.*, pp. 88–93, 2016.
- [3] H. Watanabe, M. Bouazizi, and T. Ohtsuki, “Hate Speech on Twitter: A Pragmatic Approach to Collect Hateful and Offensive Expressions and Perform Hate Speech Detection,” *IEEE Access*, vol. 6, pp. 13825–13835, 2018.
- [4] F. Salem, “Arab Social Media Report 2017: Social Media and the Internet of Things: Towards Data-Driven Policymaking in the Arab World,” 2017.

- [5] F. Miro-Llinares and J. J. Rodriguez-Sala, "Cyber hate speech on twitter: Analyzing disruptive events from social media to build a violent communication and hate speech taxonomy," *Int. J. Des. Nat. Ecodynamics*, vol. 11, no. 3, pp. 406–415, 2016.
- [6] A. Brown, "What is hate speech? Part 1: The Myth of Hate," *Law Philos.*, vol. 36, no. 4, pp. 419–468, 2017.
- [7] M. Y. Anis and U. S. Maret, "Hatespeech in Arabic Language," in *International Conference on Media Studies*, 2017, no. September.
- [8] N. Chetty and S. Alathur, "Hate speech review in the context of online social networks," *Aggress. Violent Behav.*, vol. 40, no. May, pp. 108–118, 2018.
- [9] A. Jha and R. Mamidi, "When does a compliment become sexist? Analysis and classification of ambivalent sexism using twitter data," *Proc. Second Work. NLP Comput. Soc. Sci.*, pp. 7–16, 2017.
- [10] N. Albadi, M. Kurdi, and S. Mishra, "Are they Our Brothers? Analysis and Detection of Religious Hate Speech in the Arabic Twittersphere," *2018 IEEE/ACM Int. Conf. Adv. Soc. Networks Anal. Min.*, pp. 69–76, 2018.
- [11] T. Gelashvili and K. A. Nowak, "Hate Speech on Social Media," Lund University, 2018.
- [12] P. Fortuna and S. Nunes, "A Survey on Automatic Detection of Hate Speech in Text," *ACM Comput. Surv.*, vol. 51, no. 4, pp. 1–30, 2018.
- [13] R. Irfan et al., "A survey on text mining in social networks," *Knowl. Eng. Rev.*, vol. 30, no. 2, pp. 157–170, 2015.
- [14] J. Hirschberg and C. D. Manning, "Advances in natural language processing," *Science (80-.)*, vol. 349, no. 6245, p. 261 LP-266, Jul. 2015.
- [15] S. Sun, C. Luo, and J. Chen, "A review of natural language processing techniques for opinion mining systems," vol. 36, no. November 2016. 2017.
- [16] M. M. Najeeb, A. A. Abdelkader, and M. B. Al-Zghoul, "Arabic Natural Language Processing Laboratory serving Islamic Sciences," *Int. J. Adv. Comput. Sci. Appl.*, vol. 5, no. 3, pp. 114–117, 2014.
- [17] S. Khoja and R. Garside, "Stemming arabic text," Lancaster, UK, *Comput. Dep. Lancaster Univ.*, 1999.
- [18] S. H. Ghwanmeh, G. Kanaan, R. Al-Shalabi, and S. Rabab'ah, "Enhanced Algorithm for Extracting the Root of Arabic Words," *2009 Sixth Int. Conf. Comput. Graph. Imaging Vis.*, pp. 388–391, 2009.
- [19] M. Boudchiche, A. Mazroui, M. Ould Abdallahi Ould Bebah, A. Lakhouaja, and A. Boudlal, "AlKhalil Morpho Sys 2: A robust Arabic morpho-syntactic analyzer," *J. King Saud Univ. - Comput. Inf. Sci.*, vol. 29, no. 2, pp. 141–146, 2017.
- [20] A. Alshutayri and E. Atwell, "Creating an Arabic Dialect Text Corpus by Exploring Twitter, Facebook, and Online Newspapers," no. May, 2018.
- [21] A. Goswami and A. Kumar, "A survey of event detection techniques in online social networks," *Soc. Netw. Anal. Min.*, vol. 6, no. 1, pp. 1–25, 2016.
- [22] A. Assiri, A. Emam, and H. Al-Dossari, "Towards enhancement of a lexicon-based approach for Saudi dialect sentiment analysis," *J. Inf. Sci.*, vol. 44, no. 2, pp. 184–202, 2018.

- [23] M. Wiegand, J. Ruppenhofer, A. Schmidt, and C. Greenberg, "Inducing a Lexicon of Abusive Words – a Feature-Based Approach," in Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers), 2018, pp. 1046–1056.
- [24] N. D. Gitari, Z. Zuping, H. Damien, and J. Long, "A lexicon-based approach for hate speech detection," *Int. J. Multimed. Ubiquitous Eng.*, vol. 10, no. 4, pp. 215–230, 2015.
- [25] S. George K and S. Joseph, "Text Classification by Augmenting Bag of Words (BOW) Representation with Co-occurrence Feature," *IOSR J. Comput. Eng.*, vol. 16, no. 1, pp. 34–38, 2014.
- [26] C.-F. Tsai, "Bag-of-Words Representation in Image Annotation: A Review," *ISRN Artif. Intell.*, vol. 2012, pp. 1–19, 2012.
- [27] P. Burnap and M. L. Williams, "Us and them: identifying cyber hate on Twitter across multiple protected characteristics," *EPJ Data Sci.*, vol. 5, no. 1, 2016.
- [28] G. Xiang, B. Fan, L. Wang, J. Hong, and C. Rose, "Detecting offensive tweets via topical feature discovery over a large scale twitter corpus," *Proc. 21st ACM Int. Conf. Inf. Knowl. Manag. - CIKM '12*, p. 1980, 2012.
- [29] J. Lilleberg, Y. Zhu, and Y. Zhang, "Support vector machines and Word2vec for text classification with semantic features," *Proc. 2015 IEEE 14th Int. Conf. Cogn. Informatics Cogn. Comput. ICCI*CC 2015*, pp. 136–140, 2015.
- [30] P. Burnap and M. L. Williams, "Cyber Hate Speech on Twitter: An Application of Machine Classification and Statistical Modeling for Policy and Decision Making," *Policy & Internet*, vol. 7, no. 2, pp. 223–242, 2015.
- [31] T. Hua, F. Chen, L. Zhao, C.-T. Lu, and N. Ramakrishnan, "STED: semi-supervised targeted-interest event detection in twitter," in Proceedings of the 19th ACM SIGKDD international conference on Knowledge discovery and data mining, 2013, pp. 1466–1469.
- [32] R. Pandarachalil, S. Sendhilkumar, and G. S. Mahalakshmi, "Twitter Sentiment Analysis for Large-Scale Data: An Unsupervised Approach," *Cognit. Comput.*, vol. 7, no. 2, pp. 254–262, 2015.
- [33] M. Al-Smadi, O. Qawasmeh, M. Al-Ayyoub, Y. Jararweh, and B. Gupta, "Deep Recurrent neural network vs. support vector machine for aspect-based sentiment analysis of Arabic hotels' reviews," *J. Comput. Sci.*, vol. 27, pp. 386–393, 2018.
- [34] G. K. Pitsilis, H. Ramampiaro, and H. Langseth, "Effective hate-speech detection in Twitter data using recurrent neural networks," *Appl. Intell.*, vol. 48, no. 12, pp. 4730–4742, Dec. 2018.
- [35] Y. Chen, Y. Zhou, S. Zhu, and H. Xu, "Detecting offensive language in social media to protect adolescent online safety," in Proceedings - 2012 ASE/IEEE International Conference on Privacy, Security, Risk and Trust and 2012 ASE/IEEE International Conference on Social Computing, SocialCom/PASSAT 2012, 2012, no. December, pp. 71–80.
- [36] J. H. Park and P. Fung, "One-step and Two-step Classification for Abusive Language Detection on Twitter," in AICS Conference, 2017.
- [37] H. Chen, S. McKeever, and S. J. Delany, "Abusive text detection using neural networks," in CEUR Workshop Proceedings, 2017, vol. 2086, pp. 258–260.
- [38] S. Salawu, Y. He, and J. Lumsden, "Approaches to Automated Detection of Cyberbullying: A Survey," *IEEE Trans. Affect. Comput.*, vol. 3045, no. c, pp. 1–20, 2017.

- [39] K. Dinakar, R. Reichart, and H. Lieberman, "Modeling the detection of Textual Cyberbullying,," *Soc. Mob. Web*, vol. 11, no. 02, pp. 11–17, 2011.
- [40] V. Nahar, S. Al-Maskari, X. Li, and C. Pang, "Semi-supervised Learning for Cyberbullying Detection in Social Networks," in *Databases Theory and Applications*, 2014, pp. 160–171.
- [41] M. Di Capua, E. Di Nardo, and A. Petrosino, "Unsupervised cyber bullying detection in social networks," in *2016 23rd International Conference on Pattern Recognition (ICPR)*, 2016, pp. 432–437.
- [42] S. A. Özel, E. Saraç, S. Akdemir, and H. Aksu, "Detection of cyberbullying on social media messages in Turkish," in *2017 International Conference on Computer Science and Engineering (UBMK)*, 2017, pp. 366–370.
- [43] R. Pawar, Y. Agrawal, A. Joshi, R. Gorrepati, and R. R. Raje, "Cyberbullying Detection System with Multiple Server Configurations," *2018 IEEE Int. Conf. Electro/Information Technol.*, pp. 90–95, 2018.
- [44] B. Doosje, F. M. Moghaddam, A. W. Kruglanski, A. De Wolf, L. Mann, and A. R. Feddes, "Terrorism , radicalization and de-radicalization," *Curr. Opin. Psychol.*, vol. 11, pp. 79–84, 2016.
- [45] P. Wadhwa and M. P. S. Bhatia, "Tracking on-line radicalization using investigative data mining," in *2013 National Conference on Communications (NCC)*, 2013, pp. 1–5.
- [46] S. Agarwal and A. Sureka, "Using KNN and SVM Based One-Class Classifier for Detecting Online Radicalization on Twitter," in *International Conference on Distributed Computing and Internet Technology*, 2015, pp. 431–442.
- [47] M. Fernandez and H. Alani, "Contextual semantics for radicalisation detection on Twitter," *CEUR Workshop Proc.*, vol. 2182, 2018.
- [48] W. Warner and J. Hirschberg, "Detecting Hate Speech on the World Wide Web," no. *Lsm*, pp. 19–26, 2012.
- [49] I. Kwok and Y. Wang, "Locate the Hate: Detecting Tweets against Blacks," *Twenty-Seventh AAAI Conf. Artif. Intell.*, pp. 1621–1622, 2013.
- [50] P. Burnap and M. L. Williams, "Hate Speech, Machine Classification and Statistical Modelling of Information Flows on Twitter: Interpretation and Communication for Policy Decision Making," in *Proceedings of the Conference on the Internet, Policy & Politics*, 2014, pp. 1–18.
- [51] F. Del Vigna, A. Cimino, F. Dell'Orletta, M. Petrocchi, and M. Tesconi, "Hate me, hate me not: Hate speech detection on Facebook," *CEUR Workshop Proc.*, vol. 1816, pp. 86–95, 2017.
- [52] I. Alfina, R. Mulia, M. I. Fanany, and Y. Ekanata, "Hate speech detection in the Indonesian language: A dataset and preliminary study," *2017 Int. Conf. Adv. Comput. Sci. Inf. Syst. ICACSIS 2017*, vol. 2018–Janua, no. October, pp. 233–237, 2018.
- [53] S. Jaki and T. De Smedt, "Right-wing German Hate Speech on Twitter : Analysis and Automatic Detection," p. 29, 2018.
- [54] B. Gambäck and U. K. Sikdar, "Using Convolutional Neural Networks to Classify Hate-Speech," *Assoc. Comput. Linguist.*, no. 7491, pp. 85–90, 2017.
- [55] P. Badjatiya, S. Gupta, M. Gupta, and V. Varma, "Deep Learning for Hate Speech Detection in Tweets," in *Proceedings of the 26th International Conference on World Wide Web Companion*, 2017, pp. 759–760.

- [56] Z. Zhang, D. Robinson, and J. Tepper, "Detecting Hate Speech on Twitter Using a Convolution-GRU Based Deep Neural Network," in *ESWC 2018: The Semantic Web*, 2018, pp. 745–760.
- [57] E. A. Abozinadah, A. V Mbaziira, and J. H. Jones Jr., "Detection of abusive accounts with Arabic tweets," *Int. J. Knowl. Eng.*, vol. 1, no. 2, pp. 113–119, 2015.
- [58] E. A. Abozinadah and J. H. Jones, Jr, "Improved Micro-Blog Classification for Detecting Abusive Arabic Twitter Accounts," *Int. J. Data Min. Knowl. Manag. Process*, vol. 6, no. 6, pp. 17–28, 2016.
- [59] E. A. Abozinadah and J. H. Jones, "A Statistical Learning Approach to Detect Abusive Twitter Accounts," *Proc. Int. Conf. Comput. Data Anal. - ICCDA '17*, pp. 6–13, 2017.
- [60] H. Mubarak, K. Darwish, and W. Magdy, "Abusive Language Detection on Arabic Social Media," *Proc. First Work. Abus. Lang. Online*, pp. 52–56, 2017.
- [61] A. Alakrot, L. Murray, and N. S. Nikolov, "Dataset Construction for the Detection of Anti-Social Behaviour in Online Communication in Arabic," *Procedia Comput. Sci.*, vol. 142, pp. 174–181, 2018.
- [62] A. Alakrot, L. Murray, and N. S. Nikolov, "Towards Accurate Detection of Offensive Language in Online Communication in Arabic," *Procedia Comput. Sci.*, vol. 142, pp. 315–320, 2018.
- [63] A. A. E. M. B. N. H. Alhuzali and M. Abdul-Mageed, "Think Before Your Click: Data and Models for Adult Content in Arabic Twitter," in *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*, 2018.
- [64] B. Haidar, M. Chamoun, and A. Serhrouchni, "A Multilingual System for Cyberbullying Detection : Arabic Content Detection using Machine Learning," *Adv. Sci. Technol. Eng. Syst. J.*, vol. 2, no. 6, pp. 275–284, 2017.
- [65] A. H. Alduailej and M. B. Khan, "The challenge of cyberbullying and its automatic detection in Arabic text," in *2017 International Conference on Computer and Applications (ICCA)*, 2017, pp. 389–394.
- [66] W. Magdy, K. Darwish, and I. Weber, "#FailedRevolutions: Using Twitter to Study the Antecedents of ISIS Support," in *AAAI Spring Symposium Series*, 2016.
- [67] L. Kaati, E. Omer, N. Prucha, and A. Shrestha, "Detecting Multipliers of Jihadism on Twitter," *Proc. - 15th IEEE Int. Conf. Data Min. Work. ICDMW 2015*, pp. 954–960, 2016.
- [68] N. Djuric, J. Zhou, R. Morris, M. Grbovic, V. Radosavljevic, and N. Bhamidipati, "Hate Speech Detection with Comment Embeddings," in *Proceedings of the 24th International Conference on World Wide Web*, 2015, pp. 29–30.
- [69] S. Malmasi and M. Zampieri, "Challenges in Discriminating Profanity from Hate Speech," *J. Exp. Theor. Artif. Intell.*, vol. 30, pp. 187–202, 2018.
- [70] Z. Zhang and L. Luo, "Hate Speech Detection: A Solved Problem? The Challenging Case of Long Tail on Twitter," vol. 1, no. 0, pp. 1–5, 2018.
- [71] K. E. Abdelfatah, G. Terejanu, and A. A. Alhelbawy, "UNSUPERVISED DETECTION OF VIOLENT CONTENT IN ARABIC SOCIAL MEDIA," *Comput. Sci. Inf. Technol. (CS IT)*, pp. 1–7, 2017.
- [72] W. Yin, K. Kann, and M. Yu, "Comparative Study of CNN and RNN for Natural Language Processing," *arXiv Prepr. arXiv 1702.01923*, 2017.

PARALLEL VERIFICATION EXECUTION WITH VERIFY ALGEBRA IN A CLOUD ENVIRONMENT

Kan Luo¹ Siyuan Wang¹ An Wei² Wei Yu¹ Kai Hu¹

¹School of Computer Science and Engineering, Beihang University, Beijing, China

²China Mobile (Hangzhou) Information Technology Co.,Ltd

ABSTRACT

Soft-as-a-Service (SaaS) is a software delivery model that contains composition, development and execution on cloud platforms. And massive SaaS applications need verifying before deployed. To get the verify results of a large quantity of applications in a tolerate time, verify algebra (VA) is used to cut down the number of combinations to be verified. VA is an effective way to acquire the verify statue by using previous results. In VA, the verify result is calculated without knowing the process of verification. In this way, the verification task can be distributed to servers and executed in any order. This paper proposes method called component disassembly tree to decompose a complex SaaS application. And designs a parallel verification framework in cloud environment. The Optimization of execution is discussed. The proposed parallel schema is simulated in MapReduce.

KEYWORDS

Verification, SaaS, Components Combinations

1. INTRODUCTION

Software as a Service (SaaS) is a new way for software development and delivery based on the cloud platform. In SaaS, MTA (Multi-Tenancy Architecture) [2] is a key feature. It allows all tenants software to share the same code on the basis of configuration data stored in databases or data stores. Each element represents a unique component in the SaaS system, and a set of components represents a tenant application. Tenants customize their applications using components stored in the SaaS database according to their individual requirements. However, a tenant application can be insecure or vulnerable. For those applications, some interesting properties, such as dead-lock and safety need to verifying before deployed on the cloud. With the number of components in database increasing, the workload of verification grows up greatly. For instance, supposing there are 4 layers (GUI layer, workflow layer, service layer and data layer) has 10^5 components respectively and there will be 10^{20} ($10^5 * 10^5 * 10^5 * 10^5$) possible combinations in total. It is impossible to get 10^{20} different application verified one by one. Combinatorial verifying is new verifying technique to verifying component-based applications. It verifies combinations among components which has been verified individually.

Formal method can be applied in a cloud environment leveraging the computing power offered. We have proposed the concept, VaaS (Verification-as-a-Service) on MTA, a scalable cloud-based on-demand service that uses formal models for verification. The VaaS on MTA can verify SaaS software and address behaviour, performance, and attribute aspects of software models, while it has features of SaaS software such as automated provisioning, scalability, fault-tolerant

computing, and concurrent processing[2]. In a cloud environment, different combinations can be allocated to different processors for execution in parallel. One simple way to perform combinatorial verifying in a cloud environment is: split the verification tasks, then allocate the tasks to different servers in cloud, finally summary the results. However, this is not efficient. While computing and storage resources have increased significantly, the number of combinations to be considered is still too high. Verifying all of the combinations in a SaaS system with millions of components can consume all the resources of a cloud platform.

In our previous work, we study the rule of merging the verifying status of combinations and propose a Verify Algebra System [17] to cut down component combinations to be verified. Verify algebra is an algebraic system, which defines the five statuses of verify results and four operations of merging previous verify results. According to the combinatorial structure, our proposed VA can reduce the number of combinations to be verified by using existing verification statuses and then getting the results of unknown statuses by algebraic computation. In VA, the verify status record the verify result of a certain combination. we calculate the verify result without knowing the process of verification. And the verify results are merged according to the rules of VA and the results will not be affected by the processing order and merging order. In this way, we propose a parallel and asynchronous computing mechanisms such as MapReduce, automated redundancy and recovery management, automated resource provisioning, and automated migration for scalability.

Our contribution can be summarized as: we put forward a schema that leveraging Formal Method with the computing power offered by cloud environment to check software correctness. Further, we propose a new verification framework with VA and shared databases. All verify results are saved in shared databases. the process of verification is designed where previous verifying results are used to get unknown combinations status.

This paper is structured as follows: Section II discusses the related work; Section III introduces the component disassembly tree. Section IV and Section V discusses VA parallel execution and analysis; Section VI illustrates TA experiments using the proposed solutions; and Section VII concludes this paper

2. RELATED WORK

2.1. Verification-as-a-Service

VaaS on MTA has been designed in[1]. VaaS is an architecture that can be used to verify models, similar to SaaS, beneficial from the computing power offered in a cloud. A VaaS hosts verification software in a cloud environment, and these services can be called on demand, and can be composed to verify a software model. In VaaS, Bigraph is selected as the modelling language for illustration as it can model mobile applications. A Bigraph models can be verified by first converting it to a state model, and the state model can be verified by model-checking tools. The VaaS services combination model and execution model are also presented in[8].

In a SaaS, tenants can have their customized applications stored in the SaaS databases, and often the applications are not stored as a unit in the databases. Instead, each tenant application is decomposed into its GUIs, workflows, services, and data components, and each component is stored in the database together with components of the same kinds[9]. For example, a SaaS GUI database contains all the GUI components used by all the tenants. The MTA VaaS design follows the SaaS design, for example, it has databases to store verification software, models to be verified, and verification results; it provides customization support; etc.

Essentially, an MTA VaaS is like a SaaS except the principal task is for software verification rather than general computing. A VaaS also has following unique features:

- Only formal verification software is stored in a VaaS;
- As formal verification often involves model transformation, thus a VaaS may contains transformation software;
- A VaaS also contains software for parsing formal models; and
- A VaaS may support incremental verification where subsystems are verified before whole systems are verified. Support for incremental verification includes storing model architecture and intermediate verification results, and algorithms to select only those compositions or combinations that need to be verified.

A tenant can develop a new verification application, i.e., a tenant application, by identifying and reusing GUIs (G), Workflows (W), Services (S), and Data components (D) in the VaaS database. All selected components can be linked, and then compiled to produce executable code.

In VaaS, however, there are too many compositions to be verified. Because the number of verification tasks will grow exponentially as complexity of tenant application grows. To address this issue, verification algebra rules are introduced into VaaS Architecture.

2.2. Verify Algebra

In VA, each combination can be in one of the following five states:

- **Infeasible (X)**: some components are not permitted to be combined. For example, there would be a conflict when two components do same things but cause different results such as two GUI components, one of which paints the background BLUE but the other paint RED.
- **Failed (F)**: Verification to combination on a certain property is not passed.
- **Passed (P)**: Verification to combination on a certain property is passed.
- **Irrelevant (N)**: For some combinations which are impossible to be created, but is still feasible to be verified, so there is no need to verify these combinations.
- **Unknown (U)**: the status of a combination is certain but not currently known.

Verify algebra method is capable of reducing the number of combinations to be verified by using existing verification status and then getting the results of unknown status by algebraic computation. To be specific, the verification state of one combination on a certain property is $V(Com_1, p)$ and the state of another combination is $V(Com_2, p)$, we can determine the $V(Com_1 \cup Com_2, p)$ from $V(Com_1, p)$ and $V(Com_2, p)$. To do this, four kinds of binary operations are defined as follows:

(1) Rules for operator \otimes

One combination contains many different sub-combinations. The status of one combination is composed by merging the result of its sub-combinations. The operation \otimes merges the *passed* sub-combinations.

(2) Rules for operator \oplus

The operation \oplus merges the *failed* sub-combinations

(3) Rules for operator \ominus

the operator \ominus combines part of the proper subsets of one combination.

(4) Rules for operator \odot

The operator \odot combines all the proper subsets of one combination.

\oplus	X	P	N	U	F	\oplus	X	F	N	U	P	\ominus	X	N	F	P	U	\odot	X	N	U	F	P	
X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X
P	X	P	P	P	P	F	X	F	F	F	F	N	X	N	N	N	N	N	X	N	N	N	N	N
N	X	P	N	N	N	N	X	F	N	N	N	F	X	N	U	U	U	U	X	N	U	U	U	U
U	X	P	N	U	U	U	X	F	N	U	U	P	X	N	U	U	U	F	X	N	U	F	U	U
F	X	P	N	U	U	P	X	F	N	U	U	U	X	N	U	U	U	P	X	N	U	U	P	P

Fig. 1. Combination results

2.3. Workflow Patterns

In SaaS, components are combined by single or multi patterns. The combination patterns of components are mainly expressed by the workflow. The pattern of workflow has been discussed by Van Der Alast et al. in[13]. They classify workflow patterns into six categories, namely Basic Control Flow Patterns, Advanced Branching and Synchronization Patterns, Structural Patterns, Patterns involving Multiple Instances, State-based Patterns, and Cancellation Patterns. In the six classes of workflow patterns, five patterns can be expressed by Basic Control Patterns. Furthermore, tenant applications of multi-patterns in SaaS platform can be disassembled into multi-tiered combinations that contains only the Basic Control Patterns. For this reason, emphasis of analysing combination patterns will be put on Basic Control Patterns. There are five kinds of workflow patterns in Basic Control Patterns.

- **Sequence:** The sequence pattern is the most common pattern used to model consecutive steps in a workflow process. A component in TA is enabled after the completion of another component in the same workflow pattern process, as Fig. 2 (a).
- **Parallel Split:** Multiple components are simultaneously enabled after the completion of another component, thus allowed to be executed in parallel or in any order, as Fig. 2 (b).
- **Synchronization:** Synchronization is a point in workflow process where multiple parallel components converge into one single component, thus synchronizing execution of multiple components, as Fig. 2 (c).
- **Exclusive Choice:** One of several components is chosen based on a decision or workflow control data in pattern of Exclusive Choice, as Fig. 2 (d).
- **Simple Merge:** The completions of two or more alternative components come together without synchronization in pattern of Simple Merge. In other words, the merge will be triggered once any of incoming transitions are triggered, as Fig. 2 (e).

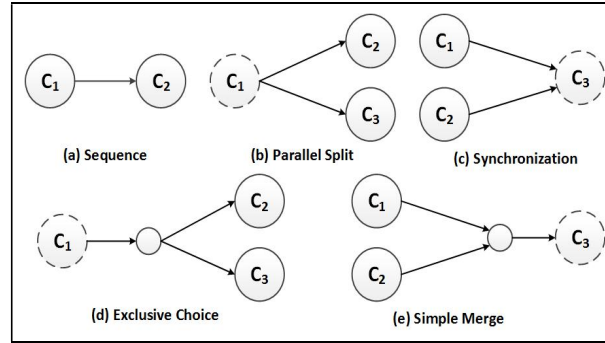


Fig. 2. Basic workflow patterns

3. COMPONENT DISASSEMBLY TREE

3.1. Abbreviation and Definition

TA: Tenant application

C: Component, a single component in SaaS databases, the basic unit in components combination. Supposing that all single components are tested or verified to be correct.

Com: Combination, constructed by more than one component.

Definition I: (Sub-Combination) If the components set Com comprises Com' , it can be claimed that combination Com' is a sub-combination of combination Com . Sub-combination is defined as $Com' \subset Com$.

Definition II: (Similar Sub-Combination) If $Com' \subset Com$ and the work flow pattern of Com' is the same as Com , It can be said that Com' is a similar *Sub-Combination* of Com . *Sub-Combination* is defined as $Com' \subset Com$.

In practical applications, most *TAs* are under hybrid patterns. For Com under hybrid patterns, it is complex to get its sub-combinations with same basic pattern and therefore, combination verification algebra can be directly used in this case. To solve this problem, a method is proposed that *TA* is disassembled and substitute by combinations with more simple patterns. In this method, disassembly means splitting *TA* into several sub-combinations and substitution means replacing atomic sub-combination with component.

The combination containing only one kind of workflow pattern is single-patterns combination. Multi-patterns combination is the components combination that contains more than one kind of Basic Control patterns. The rules of Single-pattern combination can be generalized to multi-patterns. To do this, the component disassembly tree is proposed. Component disassembly tree is a tree whose root node is *TA* itself and whose leaf nodes are components and other nodes of which are sub-tenant applications or combinations of *TA*.

The component disassembly tree of *TA* shown above is shown in Fig. 3

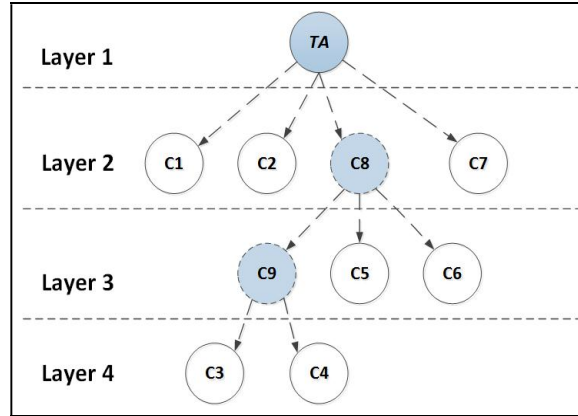


Fig. 3. Component disassembly tree of TA

A method is designed to automatically disassemble workflow graph of TA to a component disassembly tree. Supposed that the workflow graph has no circles and only has a source vertex and a destination vertex. The process of the method is illustrated as follows.

StepI: Initialize the component disassembly tree to contain a root with value TA .

StepII: Find all paths of the workflow graph using DFS.

StepIII: Search paths from head and tail to find the same nodes. Make the nodes found to the child nodes of root. And remove the nodes found from all paths. If the paths are empty, end process. otherwise go to **StepIV**.

StepIV: Add a child of root with value Com

StepV: Sort and then group the paths by first elements. For these groups, if a group only has one path and the path only has one node, make the node to be a child of Com . Else, add Com a child node with value Com , and regard this Com node as root and the group as paths, go to **StepIII**.

After component disassembly tree of tenants' application is derived, the process of combination verification algebra under hybrid patterns can be performed as follows:

Step1: Get component disassembly tree of TA .

Step2: Check if the depth of component disassembly tree is more than 2. If the depth is no more than 2, it means that TA is under single pattern and go to **Step5**. Otherwise, go to **Step3**.

Step3: Choose leaf nodes which have same parent node from bottom of the tree. It is obvious that their parent is sub-tenant application consisting of these leaf nodes and the parent node is under single pattern. Therefore, the verification status can be derived via the process represented in last section. After that, remove these leaf nodes from the tree until the parent node has no child nodes and its status is known.

Step4: Check if there is node at the bottom layer. If yes, continue **Step3**. Otherwise, decrease depth of the tree by 1 and go to **Step2**.

Step5: the verification status of TA can be derived via the process represented in last section. End the process.

4. PARALLEL VERIFICATION FRAMEWORK

When tenant application is submitted to SaaS platform, component combination verification service will use component combination algebra to verify if this application can satisfy some specific property. When this system is verifying some combination, it may need to verify its subcombination, this may need tons of computing power, therefore, we design a distributed verification framework base on component combination algebra to accelerate this process.

To use a distributed system to distribute the verification of application, we need component combination verification transaction satisfy these properties to avoid impact between different verification task or let failed verification result contaminate database of verification result.:

- Atomicity of verification task. The operations of a verification task either execute all or execute none, it can't execute only part of it. If some error occurs when execute these operations, system should rollback to the initial state.
- Consistency of verification result. These component combination verification transactions should keep system in consistence. A verification task execute at different node or time should always produce same verification result.
- Isolation of verification task. No verification task could disturb other verification task's execution. Whether these tasks are executed parallel in different node or execute serial in same node, it will always produce same result.
- Durability of verification task. After execution of a verification task, the result of this verification task will store in database, it will not be rollback by system.

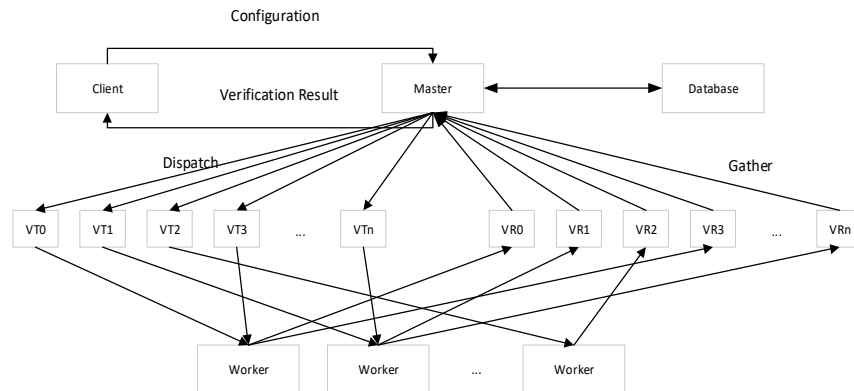


Fig. 4. Parallel verification process

When define component combination verification transaction, we need decompose component combination first. We define the smallest execution unit as component combination verification task **CT**, multiple component combination task together constituted a component combination verification transaction **T**. System will execute these component combination verification task to get component combination verification result **CR**. All component combination verification results together constituted component combination verification result **TR**. The framework of this system is show in the picture above. The definitions of these concepts are listed below.

Definition 1: A component combination verification task **CT** is a pair of combination model **M** and property **r**, it can represent as $CT = \langle M, r \rangle$.

Definition 2: A component combination verification transaction T is constitute of multiple component combination verification tasks, it can represent as $T = \{CT_1, CT_2, \dots, CT_n\}$.

Definition 3: A component combination verification result CR is a tuple of component combination model M , verification property r and verification status s . It can represent as: $CR = \langle M, r, s \rangle, s \in \{F, P\}$

Definition 4: A combination transaction's verifications result TR is a set of CR, it can represent as $TR = \{CR_1, CR_2, \dots, CR_n\}$.

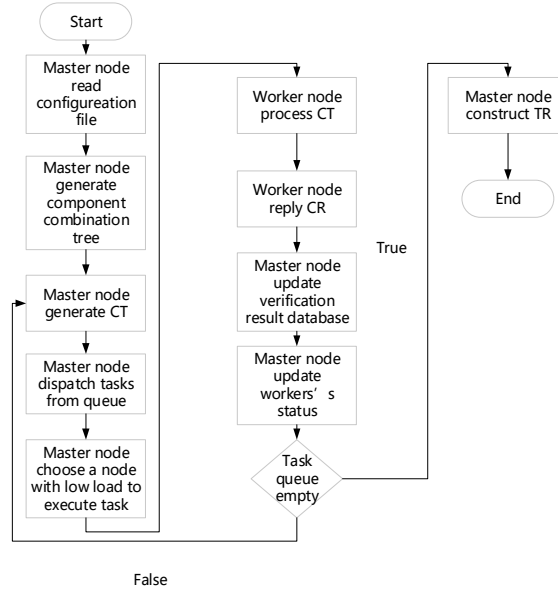


Fig. 5. VA execution flow chart

In order to decompose tenant applications, we use component combination tree to define the relation between components. After system convert user tenant application to a component combination tree, it will use this tree to generate component combination verification task and assign these tasks to different node of worker swarm to verify this application. Figure 10 shows the procedure of this algorithm:

1. Master node read tenant application configuration file and parse component combination tree from it, each node in component combination tree represent a component combination, the component combination represent by leaf can't be divide further, the component combination represent by root node is the combination of whole tenant application.
2. Master node using property need to be verified and the component represented by leaf node to construct a CT . Then master node will put these tasks into a priority queue, the lower the depth of the leaf nodes, the higher the priority.
3. Master node extract component combination verification task from priority queue and assign it to a worker node with low load. Worker node will send verification result to master node after it finish verification.
4. Master node will use verification result CR to mark the verification status of corresponding node, then delete this leaf node. Master node will also put the result into database. If the verification status of leaf node will influence the verification status of parent node, it will recursively mark the verification result of parent node, remove the subtree of parent node

and abort all verification task associated with this subtree. Then go to step 5 if the verification result of root node is confirmed, otherwise go to step 2.

Master node will use component combination result to construct a component combination verification transaction result and send it back to user.

5. ANALYSIS

Whether using distributed system or a single computer system to verify a tenant application, it's possible to encounter a scenario which it only needs to verify one submodule then produce the result of component combination verification transaction. But because of the low probability of this situation, in most cases, it's required to verify majority of component combination to complete the verification of tenant application. Let the time of produce verification task from tenant application configuration file as t_a , average execution time of verification task as t_e . Assume we use component combination tree to break a tenant application into n sub-component combination, then generate n component verification task from it. Using serial execution of verification algorithm, it will cost

$$T_s = n * (t_a + t_e) \quad (1)$$

If we use distributed component combination verification service to verify this application, assume there is k worker node. Because the master node needs to spend extra time to manage the status of worker nodes, assume the overhead of this is t_m . Then, the time use to verify this application is

$$T_d = n * t_g + \frac{n}{k} * t_e + k * t_m \quad (2)$$

From the above equation, it can be seen that the execution time of verification process depends on the number of worker node in addition to the verification transaction. When the time of node management is negligible, if the number of nodes is smaller than number of tasks, then the verification time will decrease as the number of nodes increase, if the number of nodes is larger than number of tasks, add more worker node won't bring any improvement. Due to the large number of verification task, master node needs to spend much time to distribute verification task and manage worker node status, this cannot be ignored. If we use a distributed system to verify a tenant application, adding more nodes will decrease verification time in the begging, but after certain threshold, adding more nodes will increase the time of execution. We can calculate T_d will get minimum

$$T_{d \min} = 2 * \sqrt{n * t_g * t_e} + n * t_g \quad (3)$$

$$k = \sqrt{\frac{n * t_e}{t_g}} \quad (4)$$

Compared serial execution method with the parallel execution method, when the number of verification task is large, using parallel method can significantly reduce the verification time.

6. EXPERIMENTS

The experiment is organized as follows: Firstly, verification of component combination in SaaS is simulated; Then use the method proposed in this paper to decrease the quantity of combination to

be formally verified; Finally, compare the quantities of combinations using this method with that not using this method to prove the validity and efficiency of the method in this paper.

6.1. Environment of the Simulations

(1) Hardware Environment

The simulation performs is a Hadoop distributed computing environment constructed by several virtual machines. A virtual machine is used as master node and the other seven virtual machines are used as computing nodes. All VMs have same configuration which is one CPU, 2G memory and 10G hard disk space.

(2) Software Environment

All nodes are deployed with Ubuntu Server 12.04 LTS as operating system and Hadoop 2.5.2 as run-time environment of MapReduce and Apache HBase 1.0.1.1 as distributed database. The combination verification algebra method proposed in this paper is implemented using Java programming language.

(3) Related parameters

This experiment is designed by reference to EasySaaS Architecture proposed in[17]. Components in the experiment are divided four classes: GUI components, workflow components, service components and data components. The total number of components is 100, 30% of which is GUI components, 30% of which is workflow components, 20% of which is service components, 20% of which is data components. Each tenant application consists of ten components, four GUI components, three workflow components, two service components, one data component , So there are $4.22e11$ possible combinations. The error rate is set to 0.1% and failing combinations are randomly generated at the beginning of the experiment.

Before verification starting, all tenant applications are disassembled into combinations under basic pattern, using Disassembly Tree proposed in section V. Table 1 indicates the number of combinations decomposed from applications. **Total workload** is the number of combinations to be verified for checking all applications on a certain property, and those combinations are decomposed into basic workflow pattern using component disassembled tree. After combinations status are merged with our verify algebra, the minimum combinations need to be verified is shown in **Workload after merging**.

Table 1. Heading and text fonts.

Applications number	Workload after merging	Total workload
200	159595	167400
400	311520	334800
600	459831	502200
800	603482	669600
1000	745079	837000

6.2. Simulation Experiment

This experiment is about combining different combinations on same property. In this experiment, different operations are used to be verified in parallel. The result of this experiment is shown in Fig.6. It shows the ratio of reduced time in parallel execution compared with serial execution.

From the result, it can be concluded that the ration of reducing the quantity of combinations to be verified has relation with the scale and operations described above.

When TA is on a small scale, the efficiency of combination verification method is low. But as the scale grows, the efficiency is promoted gradually. The main reason is that when verification scale grows, the probability that tenants' applications have same combinations goes higher, therefore more verify results in shared databases are reused. Meantime, as more and more sub-combinations have been verified, combination verification algebra can use the known status more times to calculate the unknow status.

For another, the efficiency of parallel verification is related to the operation type. operation with \otimes and \oplus have much less strict requirements for status of sub-combinations than that with \ominus and \odot . For operator \otimes and \oplus , status of combination can be derived as long as one of its sub-combination has a certain status. Operator \otimes and \oplus have very high efficiency. For operator \ominus , when all sub-combinations of a combination have same status, the status of the combination can be worked out. Therefore, operator \ominus has lowest efficiency. Operator \odot is a little less strict than operator \ominus but much lower than operator \otimes and \oplus .

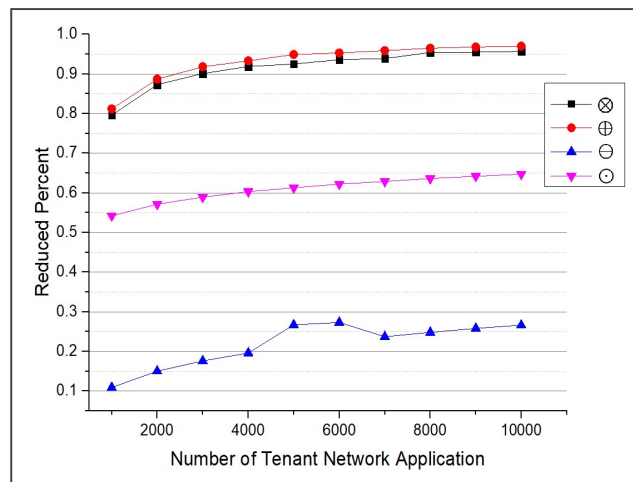


Fig. 6. Reduced percent in parallel verification

7. CONCLUSION

The VA defines the five states of verify result and operations of combining operation, provides a foundation for parallel combinatorial verifying.

In cloud platform, the computing power is utilized to check the application correctness. We have proposed VaaS on MTA, a scalable cloud-based on-demand service that uses formal models for verification.

Further, we propose a new verification framework with VA and shared databases. All verify results are saved in shared databases. the process of verification is designed where previous verifying results are used to get unknown combinations status.

By simulation experiments, it can be proved that parallel verification execution proposed in this paper is reasonable and correct and can decrease the number of the verification transactions effectively.

ACKNOWLEDGEMENTS

This work was supported by National Natural Science Foundations of China (No. 61672074,91538202). Funding of Ministry of Education and China Mo-bile MCM20160203,

REFERENCES

- [1] Yuan D M, Ren R W. Research on the SDN-Based Architecture of Space-Sky Information Network[C]//Applied Mechanics and Materials. Trans Tech Publications, 2014, 644: 2854-2856.
- [2] De Cusatis C, Cannista R, Hazard L. Managing multi-tenant services for software defined cloud data center networks[C]//Adaptive Science & Technology (ICAST), 2014 IEEE 6th International Conference on. IEEE, 2014: 1-5.
- [3] Nunes B A A, Mendonca M, Nguyen X N, et al. A survey of software-defined networking: Past, present, and future of programmable networks[J]//Communications Surveys & Tutorials, IEEE, 2014, 16(3): 1617-1634.
- [4] Open Networking Foundation. SDN architecture, version 1.1, 2016.02.
- [5] Monsanto C, Reich J, Foster N, et al. Composing software defined networks[C]//Presented as part of the 10th USENIX Symposium on Networked Systems Design and Implementation (NSDI 13). 2013: 1-13
- [6] Shin Y Y, Kang S H, Kwak J Y, et al. The study on configuration of multi-tenant networks in SDN controller[C]//Advanced Communication Technology (ICACT), 2014 16th International Conference on. IEEE, 2014: 1223-1226.
- [7] Reich J, Monsanto C, Foster N, et al. Modular SDN programming with pyretic[J]. Technical Report of USENIX, 2013.
- [8] AuYoung A, Banerjee S, Lee J, et al. Corybantic: Towards the modular composition of SDN control programs[C]//Twelfth ACM Workshop on Hot Topics in Networks (HotNets-XII), College Park, MD. 2013.
- [9] Pelle I, Lévai T, Németh F, et al. One tool to rule them all: A modular troubleshooting framework for SDN (and other) networks[C]//Proceedings of the 1st ACM SIGCOMM Symposium on Software Defined Networking Research. ACM, 2015: 24.
- [10] Dixit A, Hao F, Mukherjee S, et al. Towards an elastic distributed SDN controller[J]. ACM SIGCOMM Computer Communication Review, 2013, 43(4): 7-12.
- [11] White S A. Business process modeling notation[J]. Specification, BPMI. org, 2004.
- [12] Alves A, Arkin A, Askary S, et al. Web Services Business Process Execution Language Version 2.0[J]. Working Draft. WS-BPEL TC OASIS, May 2005.
- [13] van Der Aalst W M P, Ter Hofstede A H M, Kiepuszewski B, et al. Workflow patterns[J]. Distributed and parallel databases, 2003, 14(1): 5-51.
- [14] Al-Shaer E, Marrero W, El-Atawy A, et al. Network configuration in a box: Towards end-to-end verification of network reachability and security[C]//Network Protocols, 2009. ICNP 2009. 17th IEEE International Conference on. IEEE, 2009: 123-132.
- [15] Zakharov V A, Smelyanskii R L, Chemeritsky E V. A Formal Model and Verification Problems for Software Defined Networks[J]. Modelirovanie i Analiz Informatsionnykh Sistem [Modeling and Analysis of Information Systems], 2013, 20(6): 36-51.
- [16] ter Hofstede A H M, Orłowska M E. On the complexity of some verification problems in process control specifications[J]. The Computer Journal, 1999, 42(5): 349-359.
- [17] Tsai W T, Colbourn C J, Luo J, et al. Test algebra for combinatorial testing[C]//Automation of Software Test (AST), 2013 8th International Workshop on. IEEE, 2013: 19-25.

AUTHOR

Kan Luo, born in Hunan province, China in 1994. He received the B.E. degree in Internet of thing from the Sichuan University, China in 2016. He is currently a postgraduate at Beihang University. His researches focus on blockchain and formal verification in Institute of System in Beihang University. His direction is verification algebra in cloud environment.



Siyuan Wang, he was born in Anhui province, China. He received bachelors in computer science from Xidian University in 2017. Currently, he is a graduate student of Beihang University, major in blockchain and distributed system. He focus on distributed verification and parallel computation on software engineering.



An wei, born in 1974, graduated from Tianjin University in 1996 with a bachelor's degree in engineering, then worked in China National Software, Unisplendour Corporation Limited, Beijing Global Safety Technology Co.,Ltd and other well-known enterprises, engaged in software development, project management, consulting and other work, he is in currently engaged in China Mobile(Hangzhou) Information Technology Co.,Ltd , in digital currency product management, technology research and development, etc., for the virtual currency, block chain technology in-depth study.



Wei Yu, he was born in Neijiang, Sichuan province, China in 1993. He received the B.E. degree in computer science and technology from the Beijing Jiaotong University, China in 2016. Yu Wei is currently a postgraduate at Beihang University. His research directions are related to high performance blockchain and smart contracts.



Kai Hu is a professor at Beihang University, China. He received his PhD degree from Beihang University in 2001. From 2001 to 2004, he did the post-doctoral research at Nanyang Technological University, Singapore. Since 2004, he is the leader of the team of LDMC in the Institute of Computer Architecture (ICA), Beihang University. His research interests concern embedded real time systems and high performance computing. He has good cooperation with IRIT and INRIA Institute of France on study of AADL and synchronous languages



AUTHOR INDEX

<i>Abdullah A. Al-Shaher</i>	73
<i>Ahmad A. Al-Hajji</i>	57
<i>An Wei</i>	101
<i>Areej Al-Hassan</i>	83
<i>Fatimah M. AlSuhaibani</i>	57
<i>Hmood Al-Dossari</i>	83
<i>Ibtusam Alashoury</i>	33
<i>Junta Watanabe</i>	45
<i>Kai Hu</i>	101
<i>Kan Luo</i>	101
<i>LI Zhimin</i>	21
<i>LUO Jie</i>	21
<i>Mabroukah Amarif</i>	11 & 33
<i>Nouf S. AlHarbi</i>	57
<i>QIU Qin 'nan</i>	21
<i>Rina Komatsu</i>	01
<i>Sakeenah Ahmed</i>	11
<i>Siyuan Wang</i>	101
<i>SONG Qiaolin</i>	21
<i>Tad Gonsalves</i>	01
<i>Tad Gonsalves</i>	45
<i>Wei Yu</i>	101
<i>ZHOU Ziyang</i>	21