Journal on
Information Technologies & Communications
https://www.ictmag.vn
Volume 2022, No.1
March

TABLE OF CONTENTS

Hand Detection and Segmentation in First Person Images Using Mask R-CNN

Huy Nguyen-Sinh[1,4], Hong Le-Thi-Thu[1], Bach Nguyen-Hoang[1], Thanh Nguyen-Chi[1], Tho Chu-Thi-Quynh[2], Huyen Nguyen-Thi-Thanh[2], Hai Vu[3,4]

[1] Institute of Information Technology - AMST, Hanoi, Vietnam
[2] Department of Rehabilitation, Hanoi Medical University Hospital, Hanoi, Vietnam
[3] School of Electronics and Telecommunications, Hanoi University of Science and Technology, Hanoi, Vietnam
[4] International Research Institute MICA, Hanoi University of Science and Technology, Hanoi, Vietnam

Correspondence: Hai Vu, hai.vu@hust.edu.vn
Communication: received 03 September 2021, revised 11 October 2021, accepted 14 October 2021
DOI: 10.32913/mic-ict-research.v2022.n1.995

***Abstract*: In this work, we propose a technique to automatically detect and segment hands-on first-person images of patients in upper limb rehabilitation exercises. The aim is to automate the assessment of the patient's recovery process through rehabilitation exercises. The proposed technique includes the following steps: 1) setting up a wearable camera system and collecting upper extremity rehabilitation exercise data. After filtering, labeling the left and right hand, as well as segmenting the image area of the patient's hand, a dataset consisting of 3700 images with the name RehabHand was built. This dataset is used to train hand detection and segmentation models on first-person images. 2) conducted a survey of automatic hand detection and segmentation models using Mask-RCNN network architecture with different backbones. From the experimental architectures, the Mask-RCNN architecture with the Res2Net backbone was selected for all three tasks: hand detection, left-right hand identification, and hand segmentation. The proposed model has achieved the highest performance in the tests. To overcome the limitation on the amount of training data, we propose to use the transfer learning method along with data enhancement techniques to improve the accuracy of the model. The results of the detection of objects on the test dataset for the left hand is AP = 92.3%, the right hand AP = 91.1%. The segmentation result on the test dataset for the left hand is AP = 88.8%, the right hand being AP = 87%. These results suggest that it is possible to automatically quantify the patient's ability to use their hands during upper extremity rehabilitation.**

Keywords: *First person image, hand segmentation, transfer learning, mask R-CNN.*

I. INTRODUCTION

First Person Vision (FPV) research has received much attention in the research community recently [11]. This field of study is also known as Egocentric vision to emphasize the sensor-centered human element. FPV captures the image directly in front of the camera wearer instead of observing the entire scene like traditional surveillance cameras. Therefore, images obtained from FPV have some key advantages, such as the field of view, especially, hands and objects are in the center of the image. Image data does not include personal information (such as the face, other photo data related to biometric identification) of the person performing the operation, so it does not violate privacy and does not contain personal information in the acquired image data. Currently, wearables cameras such as GoPro Hero 9, Spectacles 2, Microsoft SenseCam used are becoming more and more popular due to the reduced cost of sensors. This has opened up a number of new application directions such as health care for the elderly, people with dementia, or in the field of sports, human-machine interaction, virtual reality, and augmentation reality. In this study, we present some initial research results on the application of FPV in the problem of upper extremity rehabilitation (especially for patients after musculoskeletal surgery, patients recovering after stroke). It is possible to use first-person imaging to evaluate the rehabilitation of patients because the field of view of hands and objects interacting during an exercise is located in the center of the image. We research and test advanced deep learning networks (Mask-RCNN) to solve such tasks as 1) detecting hand regions in images; 2) determine left or right hand; 3) hand segmentation. The results of the first task can be used as input for other problems such as tracking, hand-pose estimation. The second task was to provide information on which hand performed the exercise to compare recovery outcomes between the weak (rehabilitation) and normal hands of the same patient. The third task, hand segmentation, is used in problems that directly assess the recovery process or locate the interaction of hands and objects. Therefore, the study results can

be applied to build automated systems to quantitatively assess the recovery process for patients, helping doctors reduce costs and time and accurately estimate the process of patients' recovery for appropriate treatment.

The two main contributions of the paper include: 1) performing data collection and building a dataset for hand detection and segmentation named **RehabHand**. To our knowledge, this is the first dataset in Vietnam of patient hand images in a series of rehabilitation exercises from a first-person vision and the corresponding hand segment images with ground truth. 2) explore automatic hand segmentation and detection models using Mask-RCNN architecture [5] with different backbones. Based on the evaluation results, we recommend the Mask-RCNN model with the highest performance Res2Net [7] backbone. In the evaluation process, it is proposed to use the transfer learning method, along with some data augmentation techniques to improve the accuracy of the model.

The remainder of this paper is organized as follows: Part II describes relevant studies on FPV or egocentric vision, particularly those involving hand and object partitioning. Part III presents the construction of the dataset **RehabHand**, which is the exercises of patients with upper limb rehabilitation; The proposed method is used to detect and segment hands on the hand segments in first-person images. Part IV presents the model test settings, evaluation metrics, and test results. Finally, section V provides conclusions and directions for future work.

II. RELATED WORKS

Today, research in first-person vision focuses on recognizing actions and objects on images, especially the interaction between hands and objects. A recent survey by Bandini and colleagues [11] has comprehensively and abundantly listed related works and research directions until 2018, such as hand detection, hand segmentation; hand pose estimate; hand-object interaction activity recognition. For example, in order to recognize activities by the sensor wearer, in 2009, Ren et al. [12] laid the first foundation for the study of characterizing the sensor wearer's operation/manipulation through recognition of the object in front of him. There, the authors collected a database of 42 objects that often appear in hand operations in activities of daily living. In the proposed method, these authors used the traditional feature of SIFT and SVM classifier to detect objects. The results achieved an accuracy of only 24%. Then, Castro et al. [13] used convolutional neural networks (CNNs) to learn contextual factors combined with activity time factors to recognize activities by the sensor wearer. The authors compare the performance of Neurons networks with traditional classification methods such as k-NN, Random Forest. The proposed convolutional neural network-based technique achieves higher accuracy.

Regarding first-person databases, there have been several datasets published in the research community recently. Bambach et al. [1] introduced a new hand dataset and a deep learning model for hand detection and segmentation. Their dataset, Ego-Hands, has pixel-level annotations for hands, with two participants in each video interacting with each other. Urooj et al. [2] presented two datasets EgoYouTube – which included first-person videos containing hands in natural environments and HandOverFace with hand-held first-person images along with the appearance of arm-like areas on the image. They fine-tune RefineNet for hand segmentation and uses the hand segment result to recognize the camera wearer's hand gestures. From these surveys, it can be seen that data collected from wearable cameras and related studies published in the community about daily activities is quite common.

However, the problem of hand detection and segmentation in first-person images for rehabilitation applications has not received much attention. During these exercises, the patient wears a camera on the head (forehead) or chest. This camera captures images from a first-person perspective (sensor bearer) to capture patient hand and interactive objects data during exercises. To assess the recovery process, currently, doctors rely on visual observation of the exercise process to assess the recovery ability of the patients' hands. Therefore, once the patient's hand image is segmented during upper extremity rehabilitation exercises, it is feasible to analyze and evaluate their recovery during treatment. Likitlersuang, Jirapat et al. [4] developed a first-person video dataset, "ANS SCI" containing the hands of individuals with spinal cord injury during daily activities at home. This work proposed a deep learning model for hand detection and segmentation based on the Fast R-CNN architecture.

To the best of our knowledge, there are currently no studies on auto-detection, and hand segmentation on the first-person camera recorded images of patients during rehabilitation exercises in Vietnam. This study shows that the application of artificial intelligence can quantitatively calculate the recovery capacity of patients during rehabilitation exercises. The results will help doctors diagnose and make treatment plans for patients, especially given the data and exercises suitable to the characteristics of medical facilities in Vietnam.

III. PROPOSE METHOD

1. RehabHand dataset

The **RehabHand** dataset was collected from rehabilitation exercises of patients at Hanoi Medical University

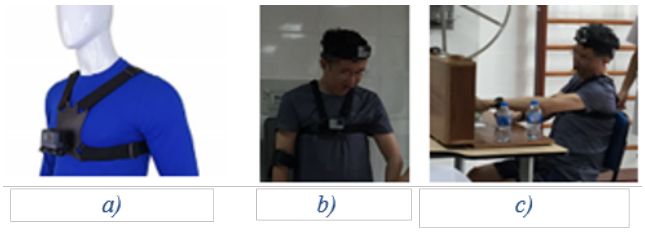


Figure 1. Set up the rehabilitation data acquisition device. a) prototype of the device; b) the patient wears the device; c) actual pictures of patients practicing exercises

Hospital. This dataset consists of frames from the first-person video captured by cameras worn by the patient on the forehead and the chest (Fig. 1). The videos are recorded with a 1080p resolution at 30 frames per second. Data were collected using GoPro Hero4 camera, San Mateo, California, USA.

The camera recorded the exercise of 15 patients performing four upper extremity rehabilitation exercises. Each patient performed each exercise five times. The content of exercises related to grasping objects in different positions is shown in Fig. 3 (next page), as follows:
Exercise 1 (Fig.3a) - practicing with the ball, pick up the round plastic balls with the hand, try to put them in the right holes;
Exercise 2 (Fig.3b) - practice with water bottles, hold a water bottle, pour water into a cup placed on the table;
Exercise 3 (Fig.3c) - practicing with the Cuboid: pick up wooden cubes with the hand, try to put them in the right holes;
Exercise 4 (Fig.3d) - practicing with cylindrical blocks, pick up the cylindrical blocks with hands, try to put them in the right holes;

From the videos obtained (videos mounted on the patient's head, chest), we decomposed into videos of different exercises, 200 video segments in total. In each of these segments, we randomly select image frames to annotate. After filtering and normalizing the data, 3700 RGB images were obtained to prepare for labeling. This selection both ensures randomness and fully represents the images of each patient and exercise. The image extracted from the patient's exercise video is saved and named the **RehabHand** dataset. The images are 1920 x 1440 pixels, with a color depth of 24 bits, and are saved as .png files. We perform annotation of the images, and each pixel in the image will correspond to a label assigned to the left-hand or right-hand, or background region. Fig. 2 illustrates the labeling results of some of the images in this dataset.

In it, the labels of the left and right hands are defined. This makes sense in practice. When detecting and partitioning the hand, it is necessary to determine which hand it is (left, right). Patients are usually weak and need one-handed rehabilitation, while the other hand can be used to evaluate and compare the recovery process (weak hand compared to normal hand).

Figure 2. Hand annotated results (left hand - green; right hand - blue)

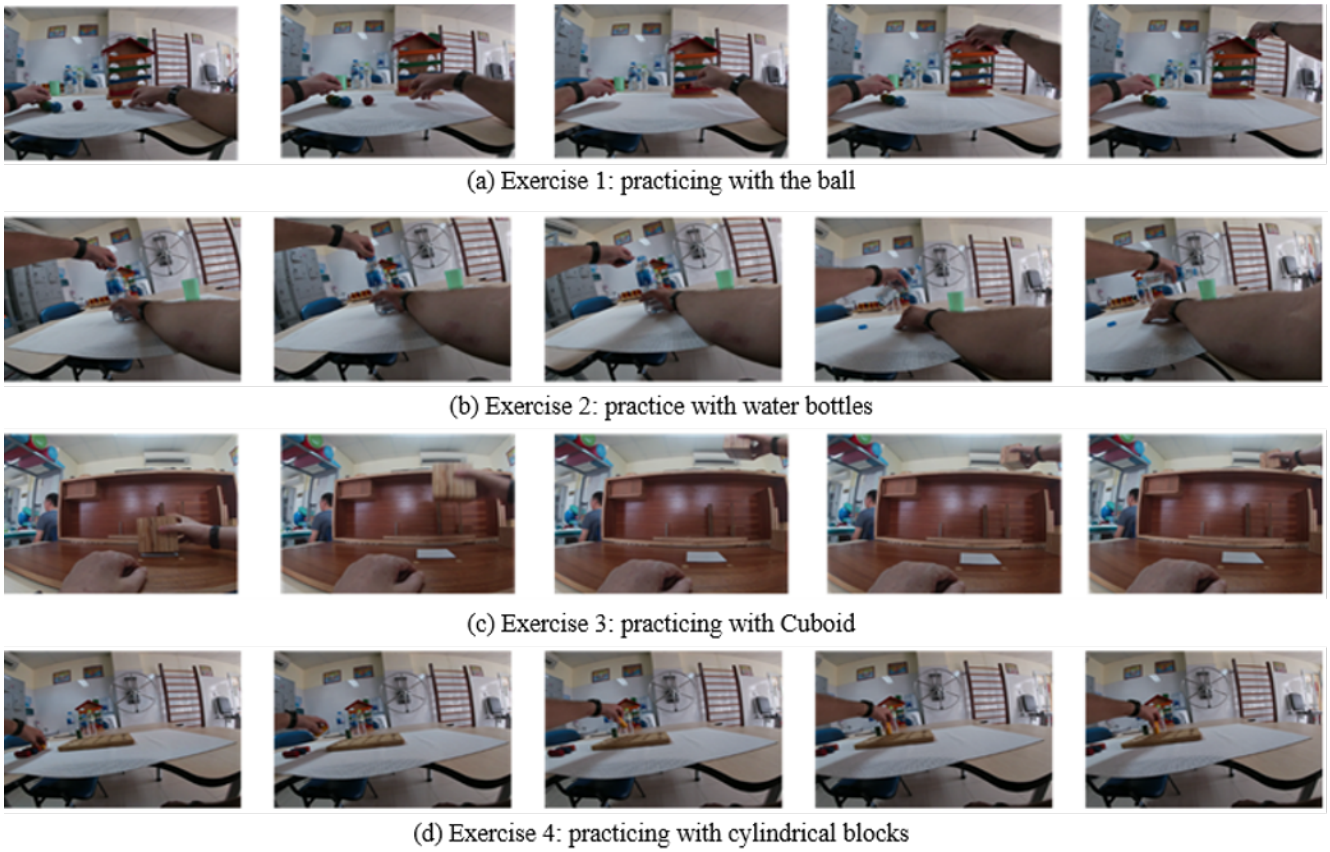
The polygon labeling for the left and right-hand areas is done manually through the VGG Image Annotator (VIA) labeling tool. Labeling results are saved in a .json format file, with the main information being: the name of the label, the filename of the labeled image, the image size, the coordinate information of the labeled areas, the coordinates of the labels. The corresponding point pair (x,y), all pairs of points (x,y) will form a polygon area surrounding the labeled arm, the id of the label (the value "7" is the label assigned to the left hand and "8" is the label assigned to the right hand). This link shares the data: https://drive.google.com/file/d/1hdeGgkTYiHvEHVEptoU8uvCbnQWjjzs/view?usp=sharing.

2. Building a Mask R-CNN network model for hand detection and segmentation

There are many ways to detect and segment hands from first-person images, such as convolutional neural networks that can detect and localize like R-CNNs. Meanwhile, single-stage networks with faster computation time only localize objects of interest such as YOLO, SSD. On the other hand, the approach with two-stage network models such as Mask R-CNN [5] has been introduced because the segmentation result needs to be more accurate than the computational time requirement.

Also, aiming for a technique that can solve all three requirements: detection of hand areas in images; left/right-hand indication; hand segmentation, we propose to build Mask R-CNN model according to the Transfer Learning method, along with evaluating the model's performance with different backbones. As a result, we selected Res2Net [7] as the backbone. Besides, we also propose some data augmentation techniques to improve the accuracy of the model.

Fig. 4 depicts the general diagram of the proposed method, including the following main parts: 1) Data Augmentation; 2) Transfer Learning with Mask R-CNN network



(a) Exercise 1: practicing with the ball

(b) Exercise 2: practice with water bottles

(c) Exercise 3: practicing with Cuboid

(d) Exercise 4: practicing with cylindrical blocks

Figure 3. Illustration of exercises in the RehabHand dataset

for hand detection and hand segmentation; 3) Save the trained model parameters; 4) Predict the hand segment on the test image. The model's outputs include circling the arms according to coordinates and length and width, hand label (left, right), and the resulting hand segmentation in the image (pixel level). These results will be input in related problems such as pose estimate, activity recognition, and measurement of movement-related features in rehabilitation exercises.

a) Data Augmentation

One of the difficulties when building automatic hand detection and segmentation models was the limited data collected for training. Additionally, because the first-person camera images are worn on the head by the patient, they can vary significantly with different environment and head movement conditions. The data augmentation step will expand the image data to cover all differences due to different environmental conditions. With data augmentation techniques, one can increase the amount of training data, thus reducing the overfitting of models. The methods of augmentation used in our work are:

- Shifting by a scale factor between -0.0625 and 0.0625: the image is shifted horizontally and vertically, with the scaling factor being a random value in the range [-0.0625 to 0.0625]. Once the random value is determined, the displacement = [tx, ty] is calculated by the scaling factor * height or width of the corresponding image.
- Increase the brightness of the image (Intensity) with a scale factor in the range [0.1, 0.3]: The contrast of the image changes with the scaling factor being a random value in the range [0.1, 0.3];
- Shift RGB color channel by random value in the range [-10, 10]: change the value of each color channel R, G, B at pixels with random increase/decrease value in the range [-10, 10];
- Shift the HSV color channel by random value: similar to each RGB color channel, the values at each pixel of the H and V channels are increased/decreased by a random value in the range [-20, 20], and for S it is in the range [-30, 30].
- We also perform image blur using Gaussian filter with mean=0, sigma=1.5, and median filter in window area w=3.

Fig. 5 illustrates some results of the applied data augmentation technique.

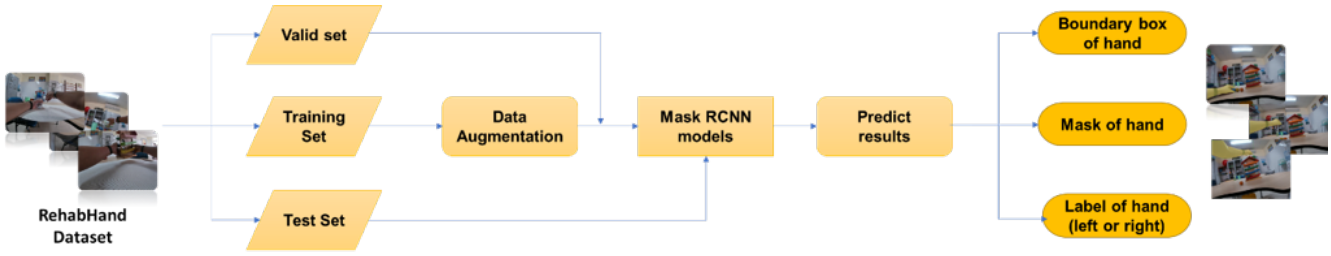


Figure 4. Diagram of the proposed method

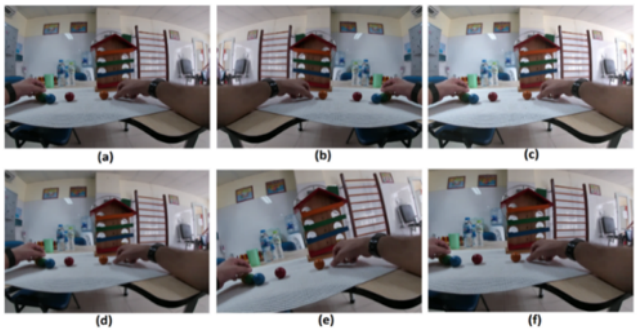


Figure 5. Some examples of data augmentation results

b) The architecture of hand detection and segmentation model

The first-person image segmentation model we use is the Mask R-CNN model. The general architecture of the Mask R-CNN network [4] is illustrated in Fig. 6. Mask-RCNN is essentially an extension of Faster R-CNN [18]. First, Faster R-CNN uses a convolutional neural network to extract feature maps from the input image. Then, the extracted image features will be passed through the RPN (Region Proposal Network) network, returning the borders around the object that can be left or right hand. The ROI pooling layer is used to bring the part of the image that was inside the bounding box in the previous step to the same size. After being brought to the same size, the image parts will be passed through a Fully Connected layer to classify objects. In addition, instead of returning the object's label and bounding box like Faster R-CNN, Mask R-CNN adds an FCN branch to show the segments (masks) of those objects. Compared with Faster R-CNN, Mask R-CNN has an improvement that replaces the RoI Pooling block with RoIAlign block, which increases the accuracy of the model more. It can be seen that Mask R-CNN is a neural network operating in two stages. Stage 1 performs feature extraction from the input image by CNN and uses RPN to give the bounding boxes of the object; Stage 2 performs the classification of the bounding boxes from phase 1, scaling the bounding boxes and creating a mask (for R-CNN Mask). The CNN network for feature extraction of the input image is called the backbone of the model, which is the image layering network that leaves out the last fully connected layer. In addition, Mask R-CNN also uses an additional network of FPN (Feature Pyramid Network) with asymmetric structure with the backbone to obtain features from many different image ratios. In this study, we evaluate the performance of the model with different backbones such as: ResNet[14], ResNeXt[15], HRNet[16], Res2Net[7], RegNet[17]. Detailed results are presented in section IV.4. After evaluation, we choose Res2Net [7] as the backbone to build the Mask R-CNN model.

The rest of our model reuses the entire architecture that the author of the Mask R-CNN model has given and uses a transfer learning method that uses parameters from the model pre-trained on a segmented dataset of objects in the wild COCO [8] as the initialization parameter for the model.

c) Model training

We use transfer learning when training hand detection and segmentation models using pre-trained model parameters on the COCO natural object segmentation dataset [8] as the initialization parameter for the model. The loss functions that are used for the models in this study are:
- L_{cls}: sum of L_{cls1} and L_{cls2}, which is the loss function of the classification. L_{cls1} is the loss calculated in the RPN network and L_{cls2} is calculated in the object detection network and generates the mask of the model objects.
- L_{bbox}: sum of L_{bbox1} and L_{bbox2}, is the loss of predicting object's position and size. L_{bbox1} is the loss calculated in the RPN network and L_{bbox2} is calculated in the object detection network of the model.
- L_{mask}: loss for predicting the object's mask, calculated by the cross-entropy function between the predicted mask and the actual mask of the object.

The final loss function used when training the Mask R-CNN network is: $L = L_{cls} + L_{bbox} + L_{mask}$

To train the image's hand detection and segmentation model, we use the Adam optimization function with momentum=0.9, initial learning rate λ=0.001, weight_decay=0.0001, batch_size=4. The weight of the net-

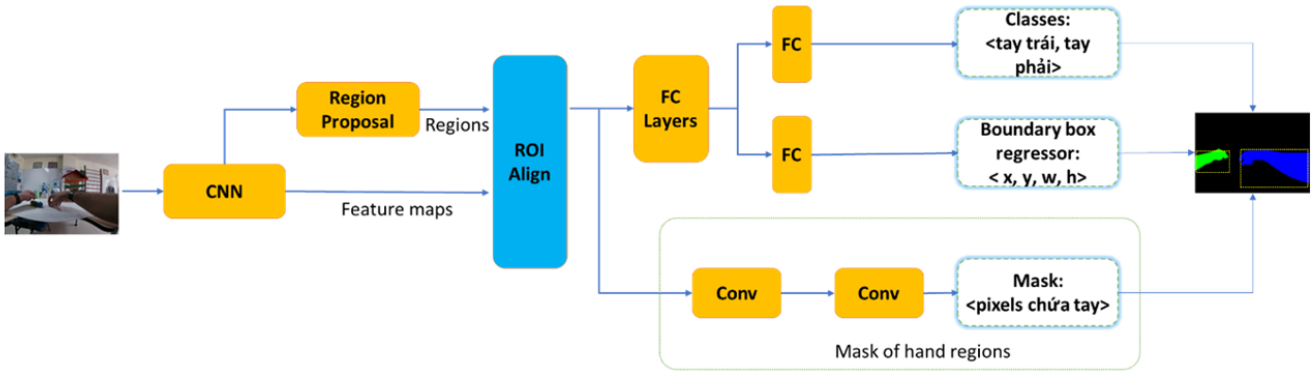


Figure 6. Mask R-CNN Network Architecture

work is pre-trained on the COCO dataset as the initialization parameter for the model. The models are then retrained on the **RehabHand** dataset. The results of the training process are illustrated in Fig. 7. It can be seen that the training converges after 24 epochs. The loss function value decreased from 1.17082 to 0.05016, with the component losses reduced as follows: loss_cls decreased from 0.36345 to 0.00448, loss_bbox 0.13667 decreased from 0.00741, loss_mask decreased from 0.64049 to 0.03752. Our models and algorithms are programmed and trained using the Python programming language and Pytorch library, Tensorflow backend on PC with GeForce GTX 1080 Ti GPU card.

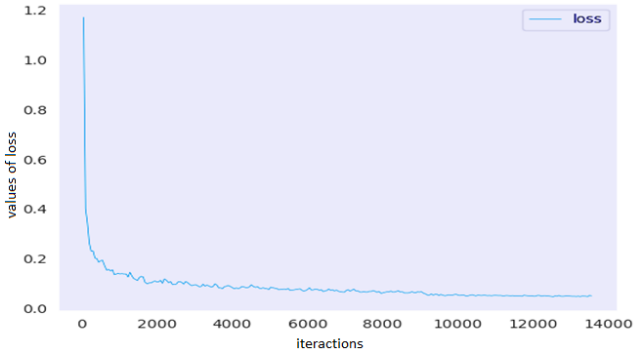


Figure 7. The value of loss function during training

IV. EXPERIEMENT AND RESULTS

1. Datasets

The **RehabHand** dataset was used with 3,700 images of 1924 x 1440 pixels that were fully and accurately labeled with a json file. We have divided this dataset into three separate subsets for training (train set), validation (valid set), and testing (test set) with the ratio of 6:2:2 respectively. There, the number of the train set images is 2220, the number of valid set images is 740, and the number of test set images is 740. Corresponding to the three image sets, there will be 3 json files labeled for the images in each set.

2. Evaluation metrics

In this work, we used AP (Average Precision) [5] as a measure to evaluate the model. The AP is an arithmetic metric that computes the inclusion of both precision and recall. AP is computed by calculating the average interpolation of the precision values over the recall values in [0,1]. We use the interpolation method at 101 recall points which is the method the authors of the dataset for segmentation of the COCO dataset proposed. We also use the AP metric proposed by these authors to evaluate the detection and segmentation models on the COCO dataset, which includes: $AP^{IoU=0.5}$ is the AP value at the threshold of IoU of 0.5, $AP^{IoU=0.75}$ is the AP value at the threshold of IoU of 0.75, and AP is the mean AP value for the threshold of IoU from 0.5 to 0.95 with each increment of 0.05

3. Performance evaluation on CNN pre-trained encoders

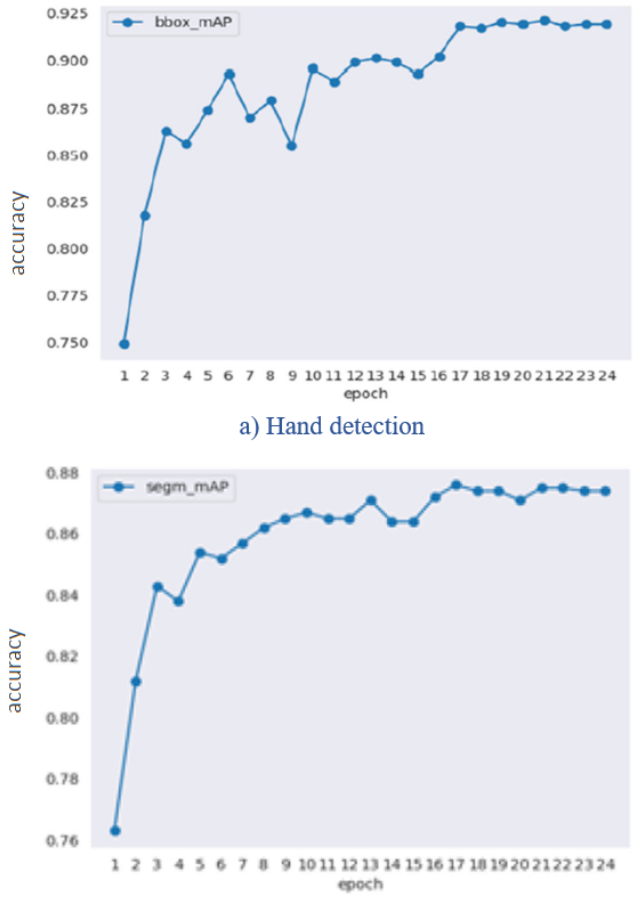
This section reports the image's hand detection and segmentation results using the Mask R-CNN model with different backbones, including ResNet, ResNeXt, HRNet, Res2Net, RegNet, and compares the performance evaluation of the model with different backbones. The experiment was conducted specifically as follows: setting up and training hand detection and segmentation models using Mask R-CNN architecture with different backbones: ResNet101 + FPN, ResNet50 + FPN, Res2Net + FPN, RegNet + FPN, ResNext + FPN and HRNet + FPN, train the models with the same training method described in section III.2. Table I shows that the Mask R-CNN model with Res2Net backbone has the highest results, with AP = 92.3% for the left-hand detection and AP = 91.3% for the right-hand detection.

TABLE I
HAND DETECTION RESULTS WITH DIFFERENT BACKBONE

Backbone	$AP^{lefthand}(\%)$	$AP^{righthand}(\%)$	$AP^{IoU=0.50}(\%)$	$AP^{IoU=0.75}(\%)$	AP(%)
ResNet101 + FPN	90.9	89.6	97.3	**96.8**	90.2
ResNet50 + FPN	90.2	88.8	97.5	96.8	89.5
Res2Net + FPN	**92.3**	**91.1**	97.7	96.6	**91.7**
RegNet + FPN	92	90	**97.8**	96.8	91
ResNext + FPN	92.1	91	96.7	**97.8**	91.7
HRNet + FPN	91.6	90.4	97	95.8	91

TABLE II
HAND SEGMENTATION RESULTS WITH DIFFERENT BACKBONE

Backbone	$AP^{lefthand}(\%)$	$AP^{righthand}(\%)$	$AP^{IoU=0.50}(\%)$	$AP^{IoU=0.75}(\%)$	AP(%)
ResNet101 + FPN	88.5	88.9	97.3	**96.7**	87.3
ResNet50 + FPN	88.3	86	97.5	96.9	87.2
Res2Net + FPN	**88.8**	**87**	97.7	97.2	**87.9**
RegNet + FPN	88.6	86.9	**97.8**	97.3	87.7
ResNext + FPN	88.7	86.6	97.3	**97.8**	87.7
HRNet + FPN	88.8	8.62	97	97	87.6



a) Hand detection

b) Hand segmentation

Figure 8. Hand segmentation accuracy on valid set during training

The average detection results are $AP^{IoU=0.50}$ = 97.7%, $AP^{IoU=0.75}$ = 96.6%, AP = 91.7%. Similarly, Table II also shows that the Mask R-CNN model with Res2Net backbone achieved the highest results, the left-hand segment with AP = 92.3%, the right-hand segment with AP = 88.8%, the average segmentation results are $AP^{IoU=0.50}$ = 87%, $AP^{IoU=0.75}$ = 97.7%, AP = 87.9%. Both tables show Mask R-CNN model with Res2Net backbone resulted in the highest performance. Hence we chose the Mask R-CNN architecture with Res2Net backbone as the architecture for our hand detection and segmentation model.

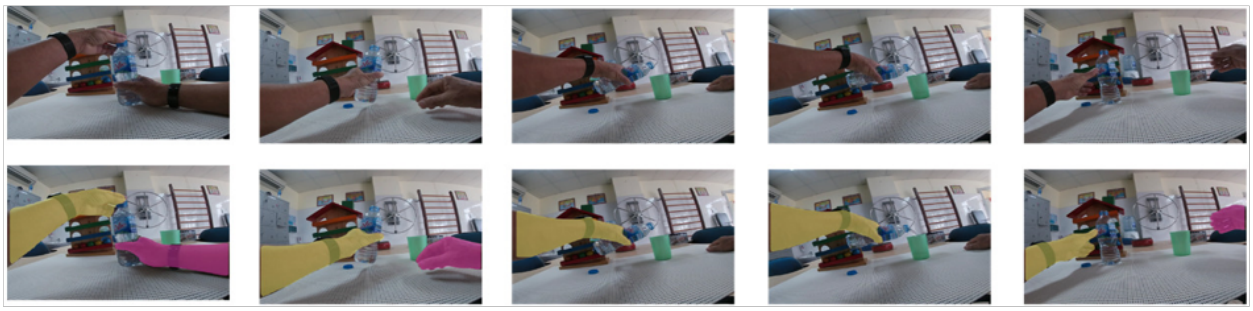
4. Results of hand detection and segmentation using Mask R-CNN architecture with Res2Net backbone

After investigating and evaluating hand detection and segmentation models using Mask R-CNN architectures with different backbones, we select Mask R-CNN with Res2Net backbone as the architecture for hand detection and segmentation model because of the highest results.

Observing the accuracy measurement values in detecting and segmenting objects on the valid set, as illustrated in Fig. 8, shows that the mean AP (mAP) value of the bounding box of the evaluated objects on the valid set increases from 0.749 to 0.919. Similarly, the mean AP value (mAP) of the segment of evaluated objects on the valid set increased from 0.763 to 0.874. Table III is the prediction results on the test data set with each class of two-handed objects.

This table shows that the left-hand detection and segmentation results are better than the right-hand. In addition,

a) Segmentation results on several frames with cylinder block exercise



b) Segmentation results on several frames with cylinder block exercise

Figure 9. Several illustrations of arm segmentation results with different exercises.
In each exercise, the top row is the original image, the bottom row is the segmentation result

TABLE III
PROPOSED MODEL EVALUATION RESULTS ON EACH CLASS

Object	Detection $AP_{bbox}(\%)$	Segmentation $AP_{segm}(\%)$
Left hand	91.1	87
Right hand	**92.3**	**88.8**

we also evaluate the accuracy of the average model on all feature classes in the image, including left-hand, right-hand, and background, and the average results on all layers are shown in Table IV.

TABLE IV
PROPOSED MODEL EVALUATION RESULTS ON ALL CLASS

Bbox/Segment	$AP^{IoU=0.50}$ (%)	$AP^{IoU=0.75}$ (%)	AP (%)
Hand Detection	97.7	96.6	91.7
Hand Segmentation	97.7	97.2	87.9

This table gives us the general results of the model: the average object detection is $AP^{IoU=0.5}$ = 97.7%, $AP^{IoU=75}$ = 96.6%, AP = 91.7% and the average object segment is $AP^{IoU=0.5}$ = 87%, $AP^{IoU=75}$ = 97.7%, AP = 87.9%. Fig. 9 visualizes some examples in rehabilitation exercises. This result shows that although the shape of the hand can change due to the posture and the way the patient performs, the resulting segmentation of the image of the hand was identified (left hand, right hand) and precisely segmented at the pixel level. Using the proposed technique, the segmentation results are almost without over/under segmentation like traditional techniques.

Discussion:

Effects of data augmentation techniques: In order to evaluate the impact of data augmentation techniques, we perform the following tests: first, install and train the data augmentation skipping model. Then install and train the model using data augmentation techniques. Finally, compare the results on the experimental dataset to assess the impact of data augmentation techniques. Table V and Table VI are comparisons of hand detection and segmentation results with or without using data augmentation techniques. These data sheets show that data augmentation contributes significantly to improving model accuracy.

TABLE V
COMPARE THE RESULTS OF THE HAND DETECTION

Data Aug.	$AP^{lefthand}$	$AP^{righthand}$	$AP^{IoU=0.50}$	$AP^{IoU=0.75}$	AP
No	91.7	90.6	96.9	96.4	90.6
Yes	**92.3**	**91.1**	**97.7**	**96.6**	**91.7**

Execution time: Mask R-CNN networks are commonly known to significantly improve computation times compared to traditional R-CNN networks, although they cannot yet reach the speeds of 1-stage networks like SSDs or YOLOs. By reducing the image resolution to 1330 x

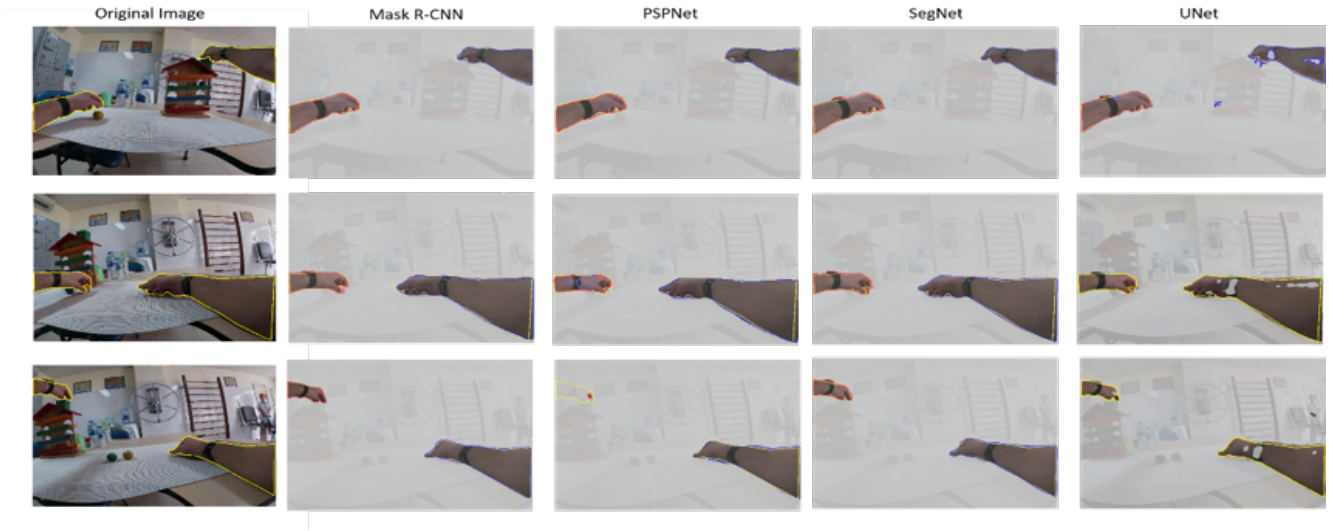


Figure 10. Illustration of some left- and right-hand segmentation results using Mask R-CNN and other comparable networks such as PSPNet, SegNet, and Unet. Column 1: Original image and segmentation results from the ground truth. Column 2: Segment results left hand (red line) and right hand (blue line) using Mask R-CNN network (alpha-blending mode from segmentation results). The yellow line is the ground-truth line. Similarly, columns 3,4,5 are the results of PSPNet, SegNet, and Unet. The color convention of the resulting segmentation contour is the same as for the R-CNN Mask result. The rows represent images at each different hand position (both left and right hand).

TABLE VI
COMPARE THE RESULTS OF THE HAND SEGMENTATION

Data Aug.	$AP^{lefthand}$	$AP^{righthand}$	$AP^{IoU=0.50}$	$AP^{IoU=0.75}$	AP
No	88.2	86.3	97.4	96.5	87.2
Yes	**88.8**	**87.0**	**97.7**	**97.2**	**87.9**

800 pixels, the calculation time can reach three fps. The image size was reduced without loss of details to achieve acceptable computation time.

Compare the results of Mask R-CNN and some popular image segmentation deep learning networks: We install three deep learning networks of object segmentation to compare with Mask R-CNN network, including SegNet[19], Unet[20], and PSPNet [21]. These models are trained and tested using the **RehabHand** dataset with the same training and testing data set as the settings with the Mask R-CNN model mentioned above. Specifically, the data set is divided into three separate subsets for training (train set), monitoring (valid set), and testing (test set) with the respective ratio of 6:2:2. Table VII shows the results of hand object segmentation with different deep learning networks. This table shows that Mask R-CNN achieved the highest accuracy with IoU for the left hand of 89.52% and the right hand of 91.87%. In addition, Fig. 10 illustrates the results of Mask R-CNN with the compared networks of SegNet, Unet and PSPNet with some image data from the **RehabHand** dataset. The illustrations show that Mask R-CNN gives the highest segmentation accuracy results compared to the compared networks. Mask R-CNN gives stable and accurate segmentation results in most cases. Meanwhile, Unet and PSPNet often suffer from under-segmentation or confusion between the left and right-hand regions. SegNet also gives good results in almost cases but often missing segmentation at the level of detail such as fingertips or arm parts.

In this paper, we have built a dataset for rehabilitation patients. These are people who are undergoing rehabilitation treatment after injury or stroke. To our knowledge, datasets similar to **RehabHand** have not been published. The patient's hand activity is different from other popular data sets such as EgoGesture [22], EgoHands [23], EGTEA [24]. These are databases collected from ordinary people with data collecting daily activities, activities in the kitchen. The performance evaluation of networks based on R-CNN architecture with databases obtained from the first-person camera can be referred to in the article [11]. The results also show that R-CNN networks have also achieved good results with data sets obtained from first-person cameras such as EgoGesture, EgoHands, and EGTEA. Although these datasets have variations in light intensity and have

TABLE VII
HAND SEGMENT WITH DIFFERENT NEURON NETWORKS

Network	**IoU**^{lefthand}**(%)**	**IoU**^{righthand}**(%)**
SegNet	80.95	83.65
Unet	75.57	79.65
PSPNet	67.58	80.02
MaskRCNN	**89.52**	**91.87**

complex backgrounds with diverse types of hand activity in daily life.

V. CONCLUSION

We have presented initial studies on artificial intelligence in the assessment of the rehabilitation process as a prerequisite for carrying out further studies. Collection and segmentation labeling, hand object detection totaling 3700 images from first-person videos of patients in upper extremity rehabilitation exercise and used this dataset for models training and pilotting. The dataset is shared within the research community. Propose a transfer learning method based on the Mask-RCNN model to detect and segment hand objects on first-person images of patients in upper limb rehabilitation exercises. After model implementation, training, and evaluation based on Mask-RCNN architecture with different recently announced backbones including ResNeXt, HRNet, Res2Net, ResNet different backbones including Resnet50, Resnet101, Res2Net, we choose Mask-RCNN architecture with Res2Net + FPN backbone as architecture due to best performance and high accuracy results in hand detection and segmentation. The left-hand detection result is AP = 92.3%, the right-hand is AP = 91.1%. The result of left-hand segmentation is AP = 88.8%, and the right-hand is AP = 87%. Although the obtained results are reliable and stable, the Mask-RCNN with backbone architecture Res2Net has to learn a rather large number of parameters and requires a lot of computational resources. The next studies are expected to research other deep learning techniques to improve the model so that the model can be deployed on systems with limited computational resources. We will also continue to study the features that can be extracted from the study results to provide quantitative results in the recovery process of patients participating in upper extremity rehabilitation exercises.

ACKNOWLEDGMENTS

This research is funded by Vietnam National Foundation for Science and Technology Development (NAFOSTED) under grant number 102.01-2017.315.

REFERENCES

[1] S. Bambach, S. Lee, D. Crandall, and C. Yu. Lending, “Detecting hands and recognizing activities in complex egocentric interactions”, *IEEE International Conference on Computer Vision (ICCV)*, 2015.

[2] Urooj, Aisha, and Ali Borji, "Analysis of hand segmentation in the wild", *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018.

[3] Likitlersuang, Jirapat, et al, "Egocentric video: a new tool for capturing hand use of individuals with spinal cord injury at home", *Journal of neuro engineering and rehabilitation*, No.16.1, p.83, 2019.

[4] He, Kaiming, et al, "Mask r-cnn", *Proceedings of the IEEE international conference on computer vision*, 2017.

[5] Lin, Tsung-Yi, et al, "Microsoft coco: Common objects in context", *European conference on computer vision. Springer, Cham*, 2014.

[6] He, Kaiming, et al, "Deep residual learning for image recognition", *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016.

[7] Xie, Saining, et al, "Aggregated residual transformations for deep neural networks", *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017.

[8] Gao, Shanghua, et al, "Res2net: A new multi-scale backbone architecture", *IEEE transactions on pattern analysis and machine intelligence*, 2019.

[9] Wang, Jingdong, et al, "Deep high-resolution representation learning for visual recognition", *IEEE transactions on pattern analysis and machine intelligence*, 2020.

[10] Radosavovic, Ilija, et al, "Designing network design spaces", *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020.

[11] Andrea Bandini, José Zariffa, “Analysis of the hands in egocentric vision: A survey”, *IEEE Transaction on Pattern Analysis and Machine Interaction*, 2020.

[12] X. Ren and M. Philipose, “Egocentric recognition of handled objects: Benchmark and analysis”, *Computer Vision and Pattern Recognition Workshops 2009, IEEE Computer Society Conference on. IEEE, 2009*, pp.1–8, 2009.

[13] D. Castro, et al, "Predicting Daily Activities from Egocentric Images Using Deep Learning", *Proceedings of the 2015 ACM International Symposium on Wearable Computers*,2015.

[14] He, Kaiming, et al, "Deep residual learning for image recognition", *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016.

[15] Xie, Saining, et al, "Aggregated residual transformations for deep neural networks", *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017.

[16] Wang, Jingdong, et al, "Deep high-resolution representation learning for visual recognition", *IEEE transactions on pattern analysis and machine intelligence*, 2020.

[17] Radosavovic, Ilija, et al, "Designing network design spaces", *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020.

[18] Shaoqing Ren, et al, “Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks”, *Proceedings of NIPS*, 2015

[19] Badrinarayanan, Vijay, Alex Kendall, and Roberto Cipolla, "Segnet: A deep convolutional encoder-decoder architecture for image segmentation", *IEEE transactions on pattern analysis and machine intelligence*, pp.2481-2495, 2017.

[20] Ronneberger, Olaf, Philipp Fischer, and Thomas Brox, "U-net: Convolutional networks for biomedical image segmentation", *International Conference on Medical image computing and computer-assisted intervention. Springer, Cham*, 2015.

[21] Zhao, Hengshuang, et al, "Pyramid scene parsing network", *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017.

[22] Yifan Zhang, Congqi Cao, Jian Cheng, Hanqing Lu, “EgoGesture: A New Dataset and Benchmark for Egocentric Hand Gesture Recognition”, *IEEE Transactions on Multimedia*, Volume 20, Issue 5, May 2018.

[23] Bambach, Sven and Lee, Stefan and Crandall, David J. and Yu, Chen, “Lending A Hand: Detecting Hands and Recognizing Activities in Complex Egocentric Interactions”, *Proceeding of The IEEE International Conference on Com-*

puter Vision (ICCV), 2015.
[24] Alireza Fathi, Xiaofeng Ren, James M. Rehg, "Learning to Recognize Objects in Egocentric Activities", *Proceeding of Computer Vision and Pattern Recognition (CVPR)*, 2011

Huy Nguyen-Sinh was born in 1976 in Thai Binh. The author graduated from the Military Technical Academy in 2000 and received his Master's degree in 2008 also from the Military Technical Academy. He is currently a researcher and PhD student at the Institute of Information Technology, AMST, Hanoi, Vietnam. The author's research area is image processing, machine vision and recognition.
Email: huyns76@gmail.com

Hong Le-Thi-Thu was born in 1980 in Thanh Hoa. The author graduated from Hanoi University of Science and Technology in 2003 and received a Master's degree in 2010 from the Military Technical Academy. He is currently a researcher and PhD student at the Institute of Information Technology, AMST, Hanoi, Vietnam. The author's research area is image processing, machine vision and recognition.
Email: lethithuhong1302@gmail.com

Bach Nguyen-Hoang was born in 1984 in Thai Binh. The author graduated from the Military Technical Academy in 2007 and received a Master's degree in 2020 also from the Military Technical Academy. He is currently a researcher at the Institute of Information Technology, AMST, Hanoi, Vietnam. The author's research area is image processing, machine vision and recognition.
Email: nhbach2203@gmail.com

Thanh Nguyen-Chi received the Ph.D. degree in Computer Science from Nagaoka University of Technology, Japan in 2012. He is currently a researcher at the Institute of Information Technology, AMST, Hanoi, Vietnam. His research interest includes deep learning, computer vision, medical image analysis, and natural language processing.
Email: thanhnc80@gmail.com

Tho Chu-Thi-Quynh was born in 1989 in Hanoi. The author graduated as a General Doctor in 2013, graduated as a Resident Doctor of Rehabilitation in 2017 at Hanoi Medical University. Currently, the author is a Lecturer at the Department of Rehabilitation at Hanoi Medical University, a doctor at the Rehabilitation Department of Hanoi Medical University Hospital. The author's research areas of interest include diagnostic-assisted medical image analysis, computer vision, and identification.
Email: chuthiquynhtho.hmu@gmail.com

Huyen Nguyen-Thi-Thanh was born in 1973 in Hai Phong, graduated as a General Doctor from Hanoi Medical University in 1996, received the degree of Resident Doctor, Level I Specialist in Rehabilitation in 2000, received Doctor of Medicine Level II majoring in Rehabilitation in 2019. Currently Head of Rehabilitation Department, Hanoi Medical University Hospital, lecturer in Rehabilitation Department, Hanoi Medical University. Member of Vietnam Rehabilitation Association. Member of the International Association for the Study of Pain (IASP). The author's research area is: rehabilitation in neurological, musculoskeletal, surgery, geriatric and pain treatment.
Email: huyen.rehab@gmail.com

Hai Vu received a PhD in Computer Science from Osaka University, Japan, 2009. Currently, the author is a lecturer at the Institute of Electronics and Telecommunications, and a research fellow in the Department of Vision. computer, MICA Institute of International Studies, Hanoi University of Science and Technology. The author's research areas of interest include diagnostic-assisted medical image analysis, computer vision, and identification.
Email: hai.vu@hust.edu.vn

A Fast Algorithm for Privacy-preserving Utility Mining

Duc Nguyen[1,2], Bac Le[1,2]

[1]Faculty of Information Technology, University of Science, Ho Chi Minh City, Viet Nam
[2]Vietnam National University, Ho Chi Minh City, Vietnam

Correspondence: Duc Nguyen, nnduc@fit.hcmus.edu.vn
Communication: received 15 December 2021, revised 22 January 2022, accepted 01 March 222
DOI: 10.32913/mic-ict-research.v2022.n1.1026

***Abstract*: Utility mining (UM) is an efficient technique for data mining which aim to discover critical patterns from various types of database. However, mining data can reveal sensitive information of individuals. Privacy preserving utility mining (PPUM) emerges as an important research topic in recent years. In the past, integer programming approach was developed to hide sensitive knowledge in a database. This approach required a significant amount of time for preprocessing and formulating a constraint satisfaction problem (CSP). To address this problem, we proposed a new algorithm based on a hash data structure which performs more quickly in itemsets filtering and problem modeling. Experiment evaluations are conducted on real world and synthetic datasets.**

Keywords: *Hash, privacy, sanitization, utility mining*

I. Introduction

Data mining is the process of discovering interesting patterns in a massive and complicated data collection. Those patterns can be used to make the world a more prosperous and secure place to live. In truth, data mining is a double-edged sword. It can be used to improve our lives through personal advertising, improved medicine and health care, reduced crime, and many more ways. At the same time, however, it can also be used to discover confidential information, to target us with unwanted advertising, and furthermore, to control our behavior through social networks. The debate between the benefits and privacy considerations of data mining are apparent in the discussions [1].

The above issues make Privacy-Preserving Data Mining (PPDM) a critical research problem. The first work was published by Agrawal et al. [2]. Since then, several methods has been proposed to hide patterns in binary datasets [3, 4]. The most common approach of PPDM is hiding sensitive information by perturbing the original database. However, it will cause several side effects such as missing non-sensitive patterns and introducing redundant information. Sanitizing data and hiding sensitive information is a NP-hard optimization problem [2, 5].

Utility pattern mining is a high practical technique in data mining that can analyze relationships between itemsets in transactional database [6]. It allows revealing itemsets yielding high profits by considering both unit profits of items and their quantities in transaction. This method can also cause privacy concerns since hidden pattern found by mining process may implicitly reveal private information [7]. For this reason, various approaches for privacy preserving utility mining (PPUM) were proposed [8]. In the majority of published studies, PPUM conceals confidential information, perturb the original database by removing or reducing quantities of certain items in transactions. The first strategies was introduced by Yeh and Hsu with two algorithms named HHUIF and MSICF [7]. Each algorithm find target transaction and item in each iteration by its own way then modify item's quantities to reduce the utility of sensitive high-utility itemset (SHUI). Another two algorithms was presented by Lin et al. [9, 10]. They hide the itemsets by transaction insertion and transaction deletion. Lin also proposed three evaluation measure for PPUM [11]. Although all sensitive itemsets can be hidden by those algorithms, obtained results have less generality and high rates of side effects. To solve this problem, an algorithm named PPUM-ILP was proposed by Li et al. [12]. They model hiding process as a CSP and solve it with integer linear programming (ILP) technique to change itemset states in a precise and general way. However, their publication does not specify the CSP formulation mechanism which can cause serious performance problem.

In this paper, we introduced a new algorithm which based on integer programming and hash data structures with main contributions as follows.

1) Introducing a hash structures called IT which can be used to quickly filter itemsets and formulating

constraints.
2) Proposing an efficient strategy for modeling sanitization process.
3) Providing extensive experimental results with more complicated utility generation to illustrate the disadvantages of ILP based approach.

II. Related works

1. High utility itemset mining

High-utility itemset mining (HUIM) was first represented by Chan et al. [13]. HUIM methods consider items with different profits and their quantities (or frequencies) in transactions not only present with binary values. They aim at discovering all itemsets that yield utilities larger than a user-defined threshold.

In recent years, HUIM has become a popular research topic. It was first represented by Chan et al. [13]. Later on, Yao and et al. published two algorithms with a formal model for HUIM [6]. For an effective pruning process, Liu et al. introduced the Transaction Weighted Utilization (TWU) [14] model. This model speed up the discovering process by finding High-TWU itemset (HTWUI). All above methods follow the level-wise approach, they perform unnecessary candidate generations and database scans. To overcome this issue, Lin et al. [15] designed a tree structured named HUP-tree. First, the algorithm find all 1-HTWUIs (HTWUIs contain only one item). After that, the HUP-tree is constructed for finding all HUIs. Another approach with a novel algorithm called HUI-miner was presented by Liu and Qu [16]. They used the utility-list structure for storing database information and directly mining HUIs without database scan either candidate generation. For real life problems, Lin et al. [17] introduced a new approach named potential high-utility itemsets mining (PHUIM) to discover itemsets with high-utility as well as itemsets with high existence probabilities. Besides, Lin proposed algorithms for mining HUIs with negative unit profits [18] and multiple minimum utility thresholds [19]. Stream data [20], closed HUIs [21, 22], etc. were also received attention.

2. Privacy-preserving utility mining

With the development of high-utility pattern mining, concerns about privacy especially in commercial environments have become increasingly important. The requirement of protecting sensitive information is becoming more and more urgent. In order to mitigate the privacy threats in utility pattern mining, privacy-preserving utility mining was proposed and got attention of researchers all over the world.

The first work in PPUM was introduced by Yeh et al. [7] with two algorithms, namely, HHUIF and MSICF. The main approach of these algorithms is using heuristic for finding and reducing the quantities of items in specific transactions, thereby achieving the goal of the problem. However, over scanning database in HHUIF and MSICF leads to slow executions. To alleviate this problem, Yun and Kim [23] constructed the FPUTT tree structure and used two structures to speed up the hiding process. FPUTT only need three database scans for the hiding process. Although faster than previous algorithms, FPUTT tree is not compact. During sanitization process, many conditional trees are generated, which make highly computational and memory costs.

Two other heuristic algorithms were proposed by Lin et al. [11] named MSU-MAU and MSU-MIU [11]. They designed the Maximum Sensitive Utility concept to delete items or reduce their quantities. Because the differences with traditional frequent itemset mining, Lin also proposed three evaluations for utility mining. In the other hand, meta-heuristic method also got adapted to solve PPUM problems. Two algorithms based on genetic algorithm, namely, PPUM+insert and PPUMGAT were designed by Li et al. [9, 10]. PPUMGA+insert insert artificial transaction to original database while PPUMGAT remove available transaction. However, insertion/deletion change the number of transactions and may introduce ghost itemsets.

Minimizing the negative effects of the data perturbation is still a problem that has to be solved. To address this problem, an improved algorithm of MSICF named IMSICF was proposed by Liu et al. [24]. The conflict counts of each sensitive itemset are computed dynamically during the hiding process. Although the computational cost is high, in worst cases, we still lose nearly eighty-percent non-sensitive information. For more accurate and general results, instead of using heuristic Li et al. [12] formulate the sanitization process into a CSP problem and use the CSP solution to hide sensitive data.

III. Preliminaries

In this section, some preliminaries and PPUM problem is described briefly. A quantitative transactional database D contains N transactions: $D = \{T_1, T_2, ..., T_N\}$. A transaction $T_n \in D (1 \leq n \leq N)$ includes one or more different items and their quantities in transaction. Let $I = \{i_1, i_2, \ldots, i_M\}$ denoted the M unique items of D, a transaction will be a subset of I ($T_n \subset I$). Each transaction is identified with a id: tid. Table I represents a quantitative database, values in table I are external utilities of items in I.

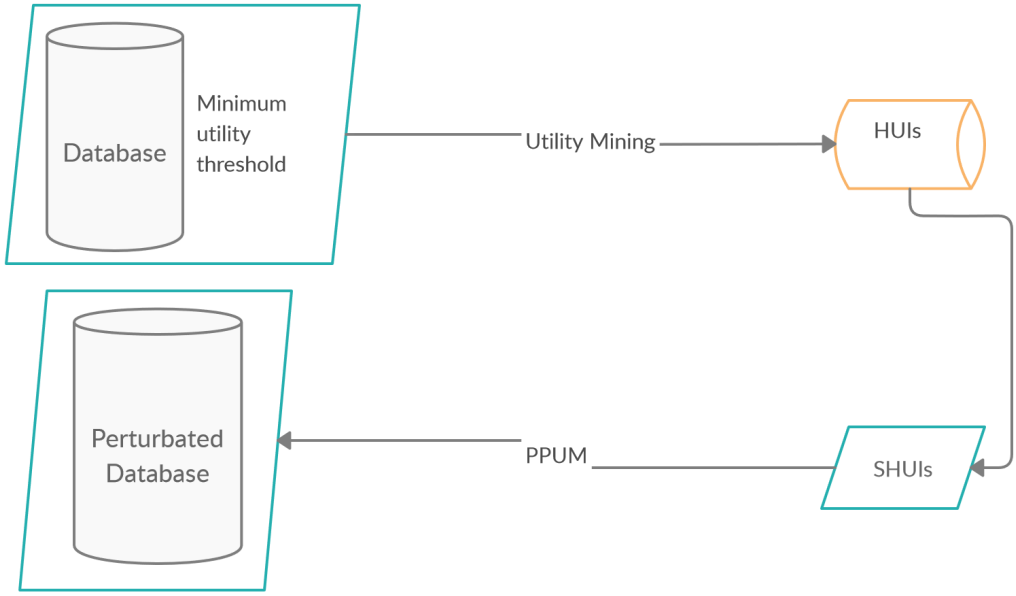


Figure 1. The general PPUM process.

TABLE I
A QUANTITATIVE TRANSACTIONAL DATABASE

tid	A	B	C	D	E
0	3	7	0	4	5
1	3	6	0	0	0
2	0	0	4	6	0
3	0	0	6	4	3
4	7	1	0	6	0
5	0	0	0	3	9
6	7	3	0	0	0
7	5	0	2	3	0
8	8	0	6	0	0
9	6	0	8	0	6

TABLE II
EXTERNAL UTILITY VALUES

Item	External Utility
A	9
B	8
C	1
D	6
E	4

Definition 1 (Internal Utility): The internal utility of item i_m in transaction T_n denoted as $\text{In}(i_m, T_n)$ is the quantity of i_m in transaction T_n.

For example, in Table I the internal utility of item A in transaction T_0 is 3: $\text{In}(A, T_0) = 3$.

Definition 2 (External Utility): The external utility of item i_m denoted as $\text{Ex}(i_m)$ represents the significant or profit of item i_m.

For example, in Table II the external utility of item B is 8: $\text{Ex}(B) = 8$.

Definition 3 (Utility of item i_m in transaction T_n): The utility of item i_m in transaction T_n denoted as $u(i_m, T_n)$ is computed by: $u(i_m, T_n) = \text{In}(i_m, T_n) \times \text{Ex}(i_m)$.

For example, the utility of item B in transaction T_0 is : $u(B, T_0) = \text{In}(B, T_0) \times \text{Ex}(D) = 7 \times 8 = 56$.

Definition 4 (The utility of itemset X in transaction T_n): The utility of itemset X in transaction T_n ($X \subseteq T_n$) denoted as $u(X, T_n)$ is the sum utility of all items of X in transaction T_n:

$$u(X, T_n) = \sum_{i_m \in X} u(i_m, T_n).$$

Note that, $u(X, T_n) = 0$ if $X \nsubseteq T_n$. Particularly, the utility of itemset $\{AD\}$ in transaction T_0: $u(\{AD\}, T_0) = u(A, T_0) + u(D, T_0) = 3 \times 9 + 4 \times 6 = 51$. Because $\{AD\} \nsubseteq T_1$ its utility in transaction T_1 $u(\{AD\}, T_1) = 0$.

Definition 5 (The utility of transaction): The utility of transaction T_n: $tu(T_n) = \sum_{i \in T_n} u(i, T_n)$.

Definition 6 (The utility of itemset X in database D): The utility of itemset X in D is the sum of utilities of X in all transactions contain X: $u(X) = \sum_{T \in D, X \subseteq T_n} u(X, T_n)$.

For example, the utility of itemset $\{AC\}$ in the database I is: $u(\{AC\}) = u(\{AC\}, T_7) + u(\{AC\}, T_8) + u(\{AC\}, T_9) = (5 \times 9 + 2 \times 1) + (8 \times 9 + 6 \times 1) + (6 \times 9 + 8 \times 1) = 47 + 78 + 62 = 187$.

Definition 7 (The minimum utility threshold): The minimum utility threshold denoted as δ, is a constant defined by the user to determine whether an itemset is valuable or not.

Definition 8 (The high-utility itemset (HUI)): An itemset X is called high-utility if its utility is greater than or equal the minimum utility δ.

Let H be the set of all high-utility itemset in D: $H = \{X \subseteq I : u(X) \geq \delta\}$. **Problem.** Given a database D, a set of HUIs H and its subset S contains sensitive high-utility itemset (SHUIs) defined by the user. PPUM problem is finding an appropriate way to modify D into the sanitized one D' that conceals all itemsets in S and minimizes the negative effects to the non-sensitive patterns in H.

A general process of PPUM is described in Figure 1

1. Side effects

All information that does not affect privacy should be retained to acquire benefits of pattern discovering process. However, hiding sensitive information cause some undesirable impact to the non-sensitive information. Three measures for the negative effects proposed by Lin [11] will be used for results evaluations. Let the original data denoted as D, the perturbed D'. H' is the set of high-utility itemsets (HUIs) in D'. Let N is the set of non-sensitive high-utility itemsets (NHUIs) in D and S is the set of SHUIs in D.

Definition 9 (Hiding Failure): Hiding Failure represents that some sensitive itemsets in D are still be found as high-utility itemsets in D'. Let HF denote the hiding failure, HF can be calculated as follows:

$$HF = \frac{|S \cap H'|}{|S|}$$

Definition 10 (Missing Cost): Missing cost indicates the lost of NSHUIs after the sanitization process. Let MC denoted the missing cost. MC can be calculated as follows:

$$MC = \frac{|N - N \cap H'|}{|N|}$$

Definition 11 (Artificial Cost): Artificial cost represents the redundant itemsets introduced by PPUM algorithms. Let AC denote the artificial cost. AC can be calculated as follows:

$$AC = \frac{|H' - H \cap H'|}{|N|}$$

In addition, in [12] Li et al. proposed a new measure named hiding cost by combining the missing cost and the artificial cost and used hiding cost to evaluate the side effects of PPUM algorithms.

Definition 12 (Hiding cost): Hiding cost denote the ratio of the number of lost NSHUIs and false HUIs in D' to the number of NSHUIs in the original database. HC can be calculated as follows:

$$HC = \frac{|N - H'|}{|N|}$$

The formula show that the hiding cost represents no information about the redundant HUIs (or the false HUIs) in D'. Therefore, we did not use the hiding cost for our evaluation.

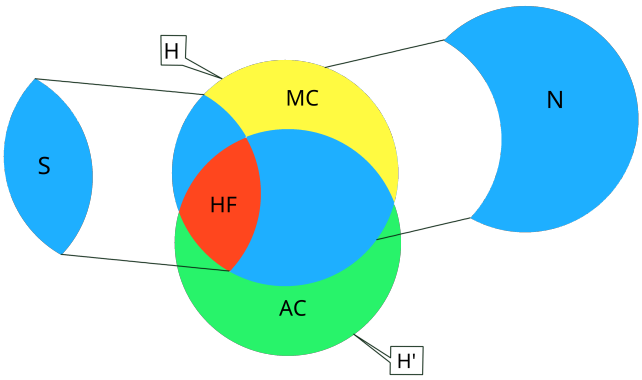


Figure 2. Three side effects of PPUM.

IV. The proposed algorithm FILP

The main idea of PPUM based on integer programming is formulation the hiding sensitive information process into a constraint satisfaction problem. The CSP has two principal objectives, i.e, hiding SHUIs and minimizing negative effects. To do that, we divide the PPUM based ILP approach into four main steps. Firstly, data structures maintain relationships between itemsets and their support transactions will be constructed. Secondly, the unaffected and redundant itemsets get removed. Thirdly, CSP is formulated and solved. Finally, the CSP solution is used to perturb the database. The whole sanitization process is described in Figure 3.

1. IT tables construction

The PPUM problem input consists of: the original database D, the set of sensitive itemsets S, the set of HUIs H and the minimum utility threshold δ. For preserving the relations between itemsets and transactions, and speeding up the preprocessing and establishing constraints processes, we designed a table structure called itemset-tidset table (IT).

Definition 13: Given a set of itemsets, the itemset-tidset table (IT) consists of 3 columns. The first column stores itemset by hash set structure. The second also uses hash set to store tids of transactions that respectively support the itemsets in first column. The third column stores the itemset utilities.

The sensitive itemset-tidset table (S-IT) is constructed by scan the original database once. Each $s_i \in I_S$ get hashed and stores in S-IT table with it utility. Transactions support

TABLE III
THE IT TABLE STRUCTURE

(Hashed) High-utility itemset	(Hashed) Tidset	Utils
a_1	t_1	u_1
a_2	t_2	u_2
a_3	t_3	u_3

s_i are found by the scan and the set of their tids gets hashed. The S-IT table is used to establish CSP and find special HUIs. With the utilization of S-IT and a prober CSP construction methods, we can reduce database scanning times in preprocessing and CSP formulation of PPUM based ILP approach and avoid handling separately each SHUI like heuristic, based methods. Non-sensitive itemset-tidset (N-IT) table also is constructed in same database scan. The only difference of N-IT and S-IT is that N-IT store information for NSHUIs instead of SHUIs. The table construction is described in Algorithm 1.

Algorithm 1: Table Construction

```
1  Input: D, the original database; H, the set of HUIs
     discovered from D; S, the set of SHUIs to be hidden
2  Output: N-IT, S-IT tables.
3  begin
4      foreach transaction T_n ∈ D do
5          foreach itemset X ∈ H do
6              if X ⊆ T_n then
7                  if X ∈ S then
8                      Insert X, U(X) into S-IT, stores tid
                         of T_n;
9                  end
10                 Insert X, U(X) into N-IT, stores tid of
                     T_n;
11             end
12         end
13     end
14     Hash stored tidsets and insert them to IT tables;
15 end
```

2. Preprocessing

Concealing SHUIs affect the NSHUIs that share same items and transactions. The consequence is losing information or introducing redundant information. As [12] suggested, filtering itemsets that get affected by perturbation make the hiding process more efficient. The proposed algorithm also used the same filtering strategy. The first phase of preprocessing is applying a filter mechanism.

Definition 14 (Filter mechanism [12]): Given the sets of sensitive itemsets S and the set of NHUIs N. Itemset $n_i \in N$ will be affected by hiding process if it shares at least one item and one transaction with any sensitive itemset $s_i \in S$

Let N' be the set of remaining NSHUIs of N after the filtering. Next step we find bound-to-lose itemsets in N'. Because such itemsets have utility lower than its sensitive superset, removing them can help avoid some conflict constraints in the CSP.

Definition 15 (Bound-to-lose itemset [22]): Given itemset X, sensitive itemset Y and the set of transactions T_X, T_Y that respectively support X, Y. X is called bound-to-lose itemset if X is a high-utility itemset and $X \subset Y \wedge T_X = T_Y$.

Let the set of remaining NHUIs is N''. Last step is removing itemsets corresponding to unnecessary constraints. In particular, if two itemset $n_{i_1}, n_{i_2} \in N''$ share same transaction set $T_{n_{i_1}}$ and $n_{i_1} \subset n_{i_2}$, n_{i_2} will be removed of N''. The whole preprocessing is described in Algorithm 2.

Algorithm 2: Preprocessing

```
1  Input: N-IT, S-IT tables
2  Output: Modified N-IT, S-IT tables. begin
3      foreach itemset n_i in N-IT do
4          L ← 0; ;
5          foreach itemset s_i in S-IT do
6              if n_i ∩ s_i ≠ ∅ and T_{n_i} ∩ T_{s_i} ≠ ∅ then
7                  if n_i ⊆ s_i and T_{n_i} = T_{s_i} then
8                      Delete n_i from N-IT;
9                  end
10             end
11             L ← L + 1;
12         end
13         if L == |S| then
14             Delete n_i from N-IT;
15         end
16     end
17     foreach itemset n_{i_2} in N-IT do
18         foreach itemset n_{i_1} in N-IT do
19             if n_{i_1} ≠ n_{i_2} and n_{i_1} ⊆ n_{i_2} and T_{n_{i_1}} = T_{n_{i_2}}
                 then
20                 Delete n_{i_2} from N-IT;
21                 break;
22             end
23         end
24     end
25 end
```

As shown in Algorithm 2, the preprocessing perform various operators to find the relation, i.e, the intersections between itemsets. Let m is the mean length of high-utility itemsets, the time complexity to find the intersection between two itemsets is $O(m^m)$. In worst cases, we need to find the intersections of $\binom{|H|}{2}$ pair of itemsets, which lead to $O\left(\binom{|H|}{2}^{m^m}\right)$. With the utilization of IT table, tidsets and itemsets are represented by hash structures. It reduced the time complexity for intersection to $O(m)$ and the whole preprocessing in the worst case to $O\left(\binom{|H|}{2}^{m}\right)$.

3. Problem formulation

The effective way to hide sensitive itemsets is changing internal utilities of item. Finding the CSP solution

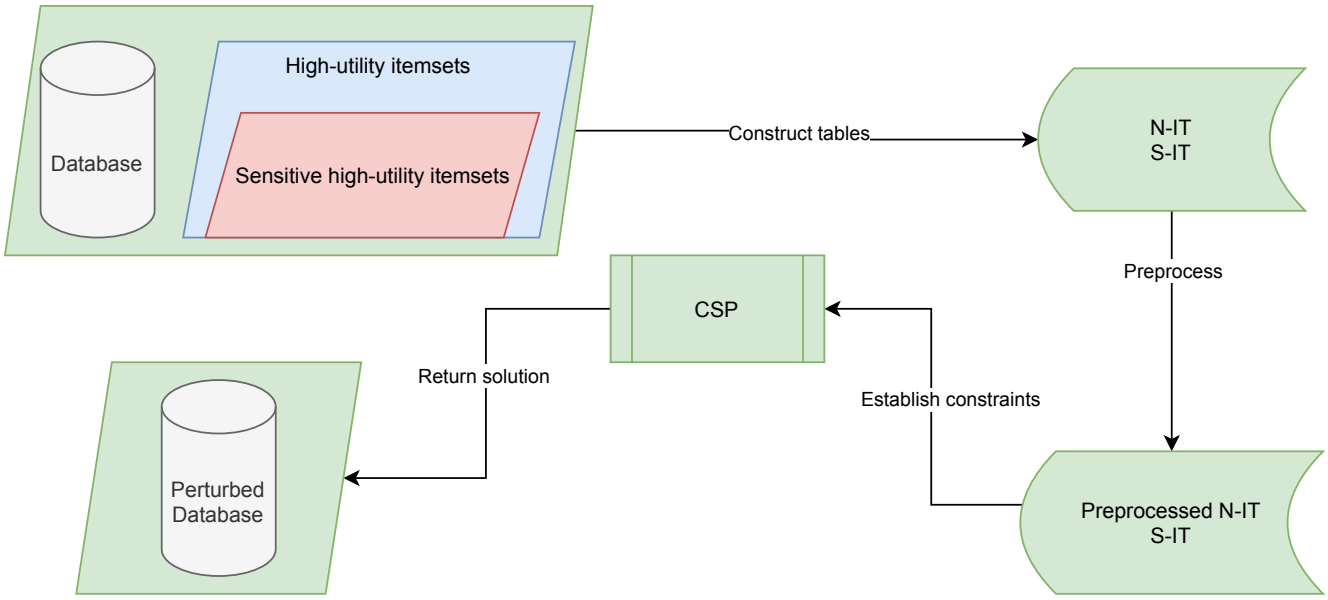


Figure 3. The PPUM based ILP approach.

is finding the optimal values for internal utilities. Those values make the sensitive high-utility itemsets become low-utility itemsets and minimize the negative impact to non-sensitive high-utility itemsets. In [12], authors did not specify the CSP formulation. The problem is a brute-force CSP formulation can lead to serious performance degradation. The proposed algorithm uses a well-designed process and utilizes the proposed data structure for faster formulating.

First, all the internal utilities of SHUIs in S is substituted by integer variables. Specifically, for each SHUI stored in S-IT, we will get the combinations of item and tid. Each combination will be used as index to indicate its utility. After this step, a hash table with variables and their utilities is constructed. Let $V = \{v_{nm} | n \in [1, N] \cap \mathbb{N}, m \in [1, M] \cap \mathbb{N}\}$, v_{nm} represents the internal utility of itemset m in transaction n. The minimum value of a variable is suggested to be 1 [12]. A SHUI X become LUI, if:

$$\sum_{T_n \in D, X \subseteq T_n} \sum_{i_m \in X} v_{nm} Ex(i_m) < \delta \tag{1}$$

In the same above step, the constraints that correspond to formula 1 also are established. Note that, for better communication with mathematical solvers, constraints should be represented by coefficient matrix. For example, let $\boldsymbol{M_S}$ be the coefficient matrix of constraints corresponding to formula 1, $\boldsymbol{v}$ be the vector of variables. To add these constraints to a solver, we simply multiply the coefficient matrix $\boldsymbol{M_S}$ with the vector of variables $\boldsymbol{v}$: $\boldsymbol{M_S v} \leq \boldsymbol{\delta}$.

Let U is the set of internal utilities of item in R:

$$U = \{u_{n'm'} | n' \in [1, N] \cap \mathbb{N}, m' \in [1, M] \cap \mathbb{N}\}$$

$u_{n'm'}$ represents two states of internal utility, i.e:

$$u_{n'm'} = \begin{cases} v_{n'm'}, & \text{if } v_{n'm'} \in V, \\ \text{In}(i_{m'}, T_{n'}), & \text{otherwise} \end{cases}$$

An itemset $Y \in R$ utility remains higher than minimum utility threshold if:

$$\sum_{T_{n'} \in D, Y \subseteq T_{n'}} \sum_{i_{m'} \in Y} u_{n'm'} Ex(i_{m'}) \geq \delta \tag{2}$$

Building constraints according to formula 2 without the hash table of variables and the N-IT table has enormous time cost. To establish the constraint of a NHUI in R, we have to find the intersection between it and each SHUI, which time complexity is $O\left(|S|^{m^m}\right)$. Besides, intersections between tidsets of SHUI and NHUI also have to be found. Let t is the average number of transactions that support a HUI, the time complexity for getting tidset intersections is $O\left(|S|^{t}\right)$. Moreover, we need to scan the database once to compute the remaining utility of the NHUI after substitution. Average time complexity for this scan (not included data loading time) is $O(m \times t)$. So the complexity for establishing the constraint of a NHUI is $O\left(|S|^{m^m} + |S|^{t} + m \times t\right)$. The complexity of establishing constraints corresponding to formula 2 is $O\left(|R|^{|S|^{m^m}+|S|^{t}+m \times t}\right)$. With the variable hash table to check whether a combination is substituted or not only has time complexity $O(1)$. Because the utilities of variables and utilities of NHUIs are stored in data structures. We

can compute the remaining utility of a NHUI only by a subtraction. The time complexity of establishing formula 2 constraints is reduced to $O(|R|^{m\times t})$. The time complexity comparison of proposed algorithm and PPUM-ILP is shown in Table IV. Let $\boldsymbol{M_n}$ be the coefficient matrix of constraints corresponding to formula 2, $\boldsymbol{r}$ be the vector of remaining utilities. These constraints can be formulated by: $\boldsymbol{M_n v} \geq \boldsymbol{\delta} - \boldsymbol{r}$.

For reducing artificial cost, the objective function is minimizing the sum of integer variables. Formally, the CSP is described as follows:

$$min \sum_{v_{nm}\in V} v_{nm} \tag{3}$$

$$s.t. \sum_{i_m\in X}\sum_{T_n\in T_X} v_{nm}Ex(i_m) \leq \delta, \qquad \forall X \in I_S \tag{4}$$

$$\sum_{i_{m'}\in Y}\sum_{T_{n'}\in T_Y} u_{n'm'}Ex(i_{m'}) \geq \delta, \qquad \forall Y \in I_R \tag{5}$$

$$v_{nm} \geq 1, \qquad \forall v_{nm} \in V \tag{6}$$

The CSP does not always arrive at a feasible solution. The proposed algorithm uses the same relaxation method introduced in [12], finding relaxation by removing constraints iteratively.

Algorithm 3: CSP Formulation

```
Input: S-IT, N-IT tables.
Output: CSP model.
begin
    Using the S-IT table create a hash table V to store
      variable positions and utilities; establish inequalities
      according to formula 4.;
    foreach itemset n_i in N-IT do
        Subtract the sum of the variable utilities in n_i
          from n_i utility.;
        Establish inequalities according to formula 5.;
    end
end
```

V. Illustrated example

In this section, the overall algorithm process is described step-by-step via an illustrated example. Returning to the database D in Table I, suppose that the minimum utility threshold δ is set to 124 and $\{AB\}$ is the sensitive itemset. The N-IT (Table V) and S-IT (Table VI) tables are first constructed. To reduce the size of problem, the itemsets in the N-IT table are preprocessed. First, the itemsets that will not be affected by the sanitization process are removed. In this case, itemsets $\{D\}$ and $\{DE\}$ are removed because they have no items in common with itemset $\{AB\}$. Itemsets $\{AC\}$ and $\{AB\}$ are not supported by any of the same transactions; thus, we delete $\{AC\}$ from the N-IT table. Itemset $\{B\}$ is not closed to $\{AB\}$, so it is eliminated to avoid conflict when establishing the constraints. Last, redundant itemsets are removed to obtain a smaller-sized problem. Itemset $\{BD\}$ is not closed to itemset $\{ABD\}$, so $\{ABD\}$ is deleted directly. Table VII shows the preprocessed N-IT table.

After preprocessing, we establish the CSP. Sensitive items of sensitive itemsets in transactions are replaced by integer variables, and constraints corresponding to formula 4 are formulated by scanning the S-IT table (Table V). Next, we scan the N-IT table (Table VII) and establish constraints for formula 5. We can compute the remaining utilities of the itemsets rapidly after substitution by subtracting variable utilities from their utilities. In our example, remaining utility of itemset $\{A\}$ after subtracting its utility from the T_0, T_1, T_4 and T_6 transactions is 171. One-hundred seventy-one is higher than minimum utility threshold (124), so there is no need to establish a constraint for $\{A\}$. In this example, the CSP does not have a feasible solution, so a relaxed solution is used to create sanitized database D' (Table VIII).

In the mining result (Table IX), sensitive itemset $\{AB\}$ is hidden; note that we lose some NSHUIs: $\{B\}$ (bound-to-lose itemset), $\{ABDE\}$ and $\{BD\}$. Furthermore, no LUI becomes a HUI. Overall, $HF = 0$, $MC = 0.3$ and $AC = 0$.

VI. Experimental results

1. Experimental environments

We performed the experiments on an Intel(R) Core(TM) i7-6700 CPU @ 3.40GHz, RAM 32GB. Two real world small datasets and a synthetic large dataset with their characteristics are displayed in Table X. The density of a dataset is measured as the average transaction length divided by the number of distinct items: Density = $\frac{AvgLen}{|I|}$. For each dataset, the external utility of each item is generated in the interval $[1, 1000] \cap \mathbb{N}$ with the log normal distribution, whereas the internal utilities of the items in the transactions are generated to be between 1 and 10 using the uniform distribution. In this paper, we used the state of the art d2HUP [25, 26] algorithm as the mining technique. The performances of the proposed algorithm were evaluated and compared with other algorithms across the datasets by execution time and side effects. The selected algorithms are as follows: PPUM-ILP, HHUIF, MSICF [7], MSU-MAU, MSU-MIU [17]; we did not compare our algorithm with PPUMGAT [9] and PPUMGAT+ [17] because these two algorithms hide information based on insertion/deletion transactions and have the different context. For better comparison all the algorithms were implemented with the high performance language Julia and the fastest solver for ILP, i.e, Gurobi 9.1.2 [27] was used to solve the CSP

TABLE IV
COMPARISON OF TIME COMPLEXITY BETWEEN PROPOSED ALGORITHM AND PPUM-ILP.

Algorithm	FILP		PPUM-ILP	
Phase	Preprocessing	Establishing formula 2	Preprocessing	Establishing formula 2
Time complexity	$O\left(\binom{\lvert H\rvert}{2}^m\right)$	$O\left(\lvert R\rvert^{m\times t}\right)$	$O\left(\binom{\lvert H\rvert}{2}^{m^m}\right)$	$O\left(\lvert R\rvert^{\lvert S\rvert^{m^m}+\lvert S\rvert^t+m\times t}\right)$

TABLE V
S-IT TABLE.

SHUI	Tidset	Utility
AB	$T_0T_1T_4T_6$	316

TABLE VI
N-IT TABLE.

NSHUI	Tidset	Utility
A	$T_0T_1T_4T_6T_7T_8T_9$	351
B	$T_0T_1T_4T_6$	136
D	$T_0T_2T_3T_4T_5T_7$	156
AC	$T_7T_8T_9$	187
AE	T_0T_9	125
AD	$T_0T_4T_7$	213
BD	T_0T_4	124
DE	$T_0T_3T_5$	134
ABD	T_0T_4	214
$ABDE$	T_0	127

TABLE VII
PREPROCESSED N-IT TABLE.

NSHUI	Tidset	Utility
A	$T_0T_1T_4T_6T_7T_8T_9$	351
AD	$T_0T_4T_7$	213
AE	T_0T_9	125
BD	T_0T_4	124
$ABDE$	T_0	127

TABLE VIII
THE SANITIZED DATABASE D'.

tid	A	B	C	D	E
0	3	5	0	4	5
1	1	1	0	0	0
2	0	0	4	6	0
3	0	0	6	4	3
4	1	1	0	6	0
5	0	0	0	3	9
6	1	1	0	0	0
7	5	0	2	3	0
8	8	0	6	0	0
9	6	0	8	0	6

(for our algorithm and PPUM-ILP). The time limit for all algorithm is set to 1200 seconds. The sensitive itemsets are randomly sampled from the set of high-utility itemsets H based on a sensitive information percentage. Execution time of an algorithm is the average of five running times. The experiments to analyze execution time were conducted under two contexts: increasing the minimum utility threshold and increasing the sensitive information percentage (SIP). Because the utility generation process is

TABLE IX
HIT TABLE FOR D'.

HUI	Tidset	Utility
A	$T_0T_1T_4T_6T_7T_8T_9$	225
D	$T_0T_2T_3T_4T_5T_7$	156
AC	$T_7T_8T_9$	187
AD	$T_0T_4T_7$	159
AE	T_0T_9	125
DE	$T_0T_3T_5$	134
ABD	T_0T_4	134

TABLE X
THE CHARACTERISTICS OF THE DATASETS.

Dataset	N	M	Density(%)
Mushrooms	8124	119	19.3
Foodmart	4141	1559	0.25
T20I6D100K	99922	929	2.22

more complicated than [12], we can see the weakness of the PPUM based ILP approach. In mushrooms datasets, the minimum utility threshold and the sensitive information percentage were set respectively to 10% and 9%. All the ILP based algorithms can not finish under the time limit (included the proposed algorithm). Because mushrooms is a dense dataset, their itemsets share many similar items. That causes many conflict constraints in CSP and moreover the CSP is harder to check whether it is feasible or not. In order to find a solution these algorithms need to iteratively remove a constraint and presolve the CSP, which makes their runtimes exceed the time limit.

2. Foodmart

The PPUM-ILP algorithm can only execute successfully in foodmart dataset. In the large and sparse T20I6D100K dataset, it failed to finish under the time limit because the computational cost for preprocessing and establishing constraints. The comparison with PPUM-ILP will be conducted separately in foodmart. Foodmart is a small and sparse datasets so that all PPUM algorithms can finish successfully the hiding. Although the length of high-utility itemsets in foodmart is small, we still gain performance advantages with the proposed IT table structure. As we can see in Figure 4 and Figure 5, all the time the proposed algorithm has lower execution time than PPUM-ILP. It also can be seen that, when the minimum utility threshold

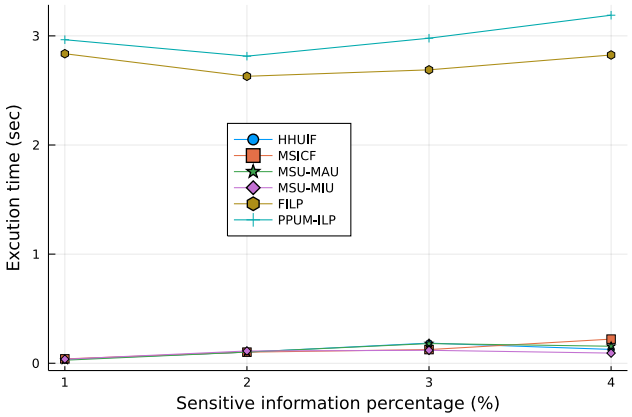


Figure 4. Execution times of algorithms in foodmart dataset under various SIP.

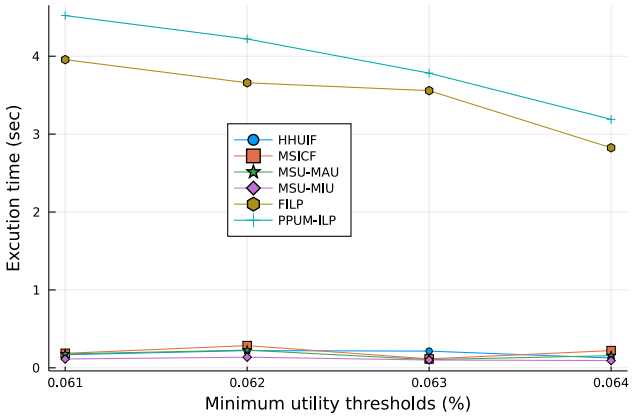


Figure 5. Execution times of algorithms in foodmart dataset under different minimum utility thresholds.

increases, the execution time decreases since the size of S become smaller. Table XI show the ability of algorithms to minimize negative effects in foodmart dataset. Since all the algorithms are able to hide all sensitive itemsets, we only compare the other two side effects. From table XI, we observed that even with the more general problem formulation the obtained solutions are not better. The reason behind this is the lack of accurate when finding a relaxation by iteratively removing constraints. As described in [12], the removed constraint of each iteration corresponds to the itemsets which has the longest length and lowest utility. But there is no mathematical proof for that constraint should be the most conflict in the CSP.

3. T20I6D100K

The execution times of algorithms in T20I6D100K is shown in Figure 6 and 7. Observing the results we see that in most cases, algorithm execution time increases as the sensitive information percentage (SIP) increases. This result

TABLE XI
SIDE EFFECTS OF ALGORITHMS IN FOODMART.

SIP	2		4	
Side effects	MC	AC	MC	AC
HHUIF	79.7%	0.0%	82.4%	0.0%
MSICF	74.5%	0.0%	81.8%	0.0%
MSU-MAU	79.7%	0.0%	82.3%	0.0%
MSU-MIU	76.3%	0.0%	80.7%	0.0%
PPUM-ILP	81.7%	2.6%	84.4%	7.1%
FILP	81.7%	2.6%	84.4%	2.8%

TABLE XII
SIDE EFFECTS OF ALGORITHMS IN T20I6D100K.

SIP	2		8	
Side effects	MC	AC	MC	AC
HHUIF	8%	0%	26.1%	0%
MSICF	8%	0%	24.2%	0%
MSU-MAU	8%	0%	26.1%	0%
MSU-MIU	4.3%	0%	20.3%	0%
FILP	0%	0.6%	8.5%	168%

is reasonable since as we increase the sensitive information percentage while maintaining a fixed minimum utility threshold, the number of sensitive high-utility itemsets increases. As a consequence, the CSP becomes harder to solve for PPUM based ILP approaches and the hiding iterations of heuristic algorithms increases. However, T20I6D100K is a sparse dataset, time for solving CSP problem is not changed much. Additionally, in worst cases, to hide a sensitive itemset, heuristic algorithms need to perform database scan for every item belong to the itemset. Too many database scans will slow down the hiding process and increase the running time. In some cases, when increases the minimum utility thresholds, the runtimes of algorithms increase. This is because the higher the minimum utility thresholds, the more frequent the itemsets appear in database and the more database scans for heuristic algorithms.

The side effects of algorithms in T20I6D100K are shown in Table XII. The FILP algorithm achieves better results for retaining non-sensitive information but in some cases it can introduce too much redundant information. Beyond that the proposed algorithm always runs faster than PPUM-ILP.

VII. Conclusion and Discussion

Data mining can reveal implicit sensitive pattern and private information in a database. Privacy concerns about data mining are becoming more and more serious. Protecting privacy in data mining, especially for utility mining is a critical problem. However, hiding sensitive patterns while retaining non-sensitive information is a NP-hard problem. In this paper, we introduced a faster PPUM method based on ILP approach with the utilization of hash structures. The conducted experiments have shown the efficiency of proposed algorithm. Besides, we still have some problems

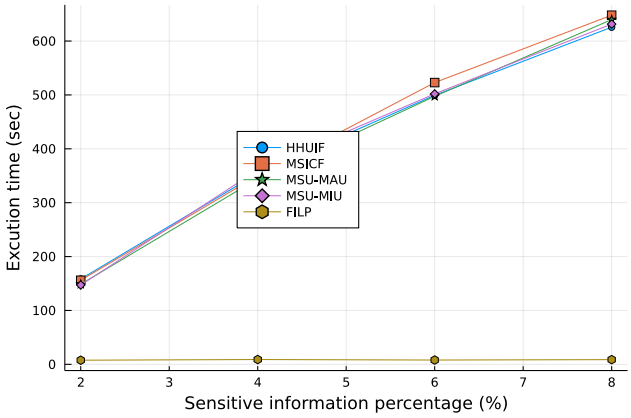


Figure 6. Execution times of algorithms in T20I6D100K dataset under various SIP.

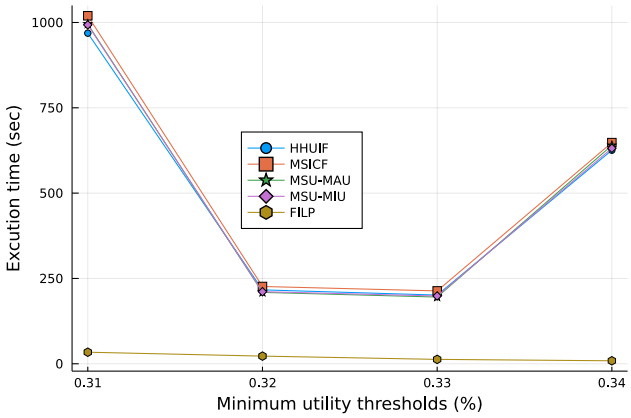


Figure 7. Execution times of algorithms in T20I6D100K dataset under different minimum utility thresholds.

with the ILP based approach. The CSP of hiding process does not have specific constraints to limit artificial cost. The time for solving the CSP is too high, so we need a better way to find a feasible solution. In addition, the preprocessing can be run asynchronously using parallel programming methods. This research is funded by University of Science, VNU-HCM under grant number CNTT 2021-05.

REFERENCES

[1] A. Mantelero and G. Vaciago, "The "Dark Side" of Big Data: Private and Public Interaction in Social Surveillance, How data collections by private entities affect governmental social control and how the EU reform on data protection responds," *Computer Law Review International*, vol. 14, pp. 161–169, 12 2013.

[2] R. Agrawal and S. Ramakrishnan, "Privacy-preserving data mining," in *Proceedings of the 2000 ACM SIGMOD International Conference on Management of Data*, ser. SIGMOD '00. New York, NY, USA: Association for Computing Machinery, 2000, pp. 439–450. [Online]. Available: https://doi.org/10.1145/342009.335438

[3] A. V. Evfimievski, S. Ramakrishnan, R. Agrawal, and J. Gehrke, "Privacy preserving mining of association rules," in *Proceedings of the Eighth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, July 23-26, 2002, Edmonton, Alberta, Canada*. ACM, 2002, pp. 217–228. [Online]. Available: https://doi.org/10.1145/775047.775080

[4] J. Vaidya and C. Clifton, "Privacy preserving association rule mining in vertically partitioned data," in *Proceedings of the Eighth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, ser. KDD '02. New York, NY, USA: Association for Computing Machinery, 2002, pp. 639–644. [Online]. Available: https://doi.org/10.1145/775047.775142

[5] V. S. Verykios, E. Bertino, I. N. Fovino, L. P. Provenza, Y. Saygin, and Y. Theodoridis, "State-of-the-art in privacy preserving data mining," *SIGMOD Record*, vol. 33, no. 1, pp. 50–57, March 2004. [Online]. Available: https://doi.org/10.1145/974121.974131

[6] H. Yao, H. J. Hamilton, and C. J. Butz, "A foundational approach to mining itemset utilities from databases," in *Proceedings of the 2004 SIAM International Conference on Data Mining*, 2004, pp. 482–486.

[7] J.-S. Yeh and P.-C. Hsu, "HHUIF and MSICF: Novel algorithms for privacy preserving utility mining," *Expert Systems with Applications*, vol. 37, no. 7, pp. 4779–4786, 2010.

[8] U. Yun and D. Kim, "Analysis of privacy preserving approaches in high utility pattern mining," in *Advances in Computer Science and Ubiquitous Computing*, J. J. J. H. Park, Y. Pan, G. Yi, and V. Loia, Eds., Singapore, 2017, pp. 883–887.

[9] C.-W. Lin, T.-P. Hong, J.-W. Wong, G.-C. Lan, and W.-Y. Lin, "A GA-based approach to hide sensitive high utility itemsets," *The Scientific World Journal*, vol. 2014, pp. 2356–6140, March 2014.

[10] J. C.-W. Lin, T.-P. Hong, P. Fournier-Viger, Q. Liu, J.-W. Wong, and J. Zhan, "Efficient hiding of confidential high-utility itemsets with minimal side effects," *Journal of Experimental & Theoretical Artificial Intelligence*, vol. 29, no. 6, pp. 1225–1245, 2017.

[11] J. C.-W. Lin, T.-Y. Wu, P. Fournier-Viger, G. Lin, J. Zhan, and M. Voznak, "Fast algorithms for hiding sensitive high-utility itemsets in privacy-preserving utility mining," *Engineering Applications of Artificial Intelligence*, vol. 55, pp. 269–284, 2016.

[12] S. Li, N. Mu, J. Le, and X. Liao, "A novel algorithm for privacy preserving utility mining based on integer linear programming," *Engineering Applications of Artificial Intelligence*, vol. 81, pp. 300–312, 2019.

[13] R. Chan, Q. Yang, and Y.-D. Shen, "Mining high utility itemsets," in *Third IEEE International Conference on Data Mining*, 2003, pp. 19–26.

[14] Y. Liu, W.-K. Liao, and A. Choudhary, "A two-phase algorithm for fast discovery of high utility itemsets," in *Advances in Knowledge Discovery and Data Mining*, T. B. Ho, D. Cheung, and H. Liu, Eds., Berlin, Heidelberg, 2005, pp. 689–695.

[15] C.-W. Lin, T.-P. Hong, and W.-H. Lu, "An effective tree structure for mining high utility itemsets," *Expert Systems with Applications*, vol. 38, no. 6, pp. 7419–7424, 2011.

[16] M. Liu and J. Qu, "Mining high utility itemsets without candidate generation," in *Proceedings of the 21st ACM International Conference on Information and Knowledge Management*, ser. CIKM '12, New York, NY, USA, 2012, pp. 55–64.

[17] J. C.-W. Lin, W. Gan, P. Fournier-Viger, T.-P. Hong, and V. S. Tseng, "Efficient algorithms for mining high-utility itemsets in uncertain databases," *Knowledge-Based Systems*, vol. 96, pp. 171–187, 2016.

[18] J. C.-W. Lin, P. Fournier-Viger, and W. Gan, "FHN: An efficient algorithm for mining high-utility itemsets with negative unit profits," *Knowledge-Based Systems*, vol. 111, pp. 283–298, 2016.
[19] J. C.-W. Lin, W. Gan, P. Fournier-Viger, and T.-P. Hong, "Mining high-utility itemsets with multiple minimum utility thresholds," in *Proceedings of the Eighth International C* Conference on Computer Science & Software Engineering*, ser. C3S2E '15, New York, NY, USA, 2015, pp. 9–17.
[20] H.-F. Li, H.-Y. Huang, and S.-Y. Lee, "Fast and memory efficient mining of high-utility itemsets from data streams: with and without negative item profits," *Knowledge and Information Systems*, vol. 28, no. 3, pp. 495–522, 2011.
[21] C. W. Wu, P. Fournier-Viger, P. S. Yu, and V. S. Tseng, "Efficient mining of a concise and lossless representation of high utility itemsets," in *2011 IEEE 11th International Conference on Data Mining*, 2011, pp. 824–833.
[22] C. Wu, P. Fournier-Viger, J. Gu, and V. S. Tseng, "Mining closed+ high utility itemsets without candidate generation," in *2015 Conference on Technologies and Applications of Artificial Intelligence (TAAI)*, 2015, pp. 187–194.
[23] U. Yun and J. Kim, "A fast perturbation algorithm using tree structure for privacy preserving utility mining," *Expert Systems with Applications*, vol. 42, no. 3, pp. 1149–1165, 2015.
[24] X. Liu, G. Chen, S. Wen, and G. Song, "An improved sanitization algorithm in privacy-preserving utility mining," *Mathematical Problems in Engineering*, vol. 2020, pp. 1–14, April 2020.
[25] J. Liu, K. Wang, and B. C. Fung, "Direct discovery of high utility itemsets without candidate generation," *2012 IEEE 12th International Conference on Data Mining*, Dec. 2012.
[26] C. Zhang, G. Almpanidis, W. Wang, and C. Liu, "An empirical evaluation of high utility itemset mining algorithms," *Expert Systems with Applications*, vol. 101, pp. 91–115, Jul. 2018.
[27] L. Gurobi Optimization, "Gurobi optimizer reference manual," 2020. [Online]. Available: http://www.gurobi.com

Duc Nguyen received B.S in 2018 and M.S in 2021 from University of Science-VNUHCM. He is currently working at Department of Computer Science, University of Science - VNUHCM. His research interests are machine learning, pattern recognition, data mining and privacy-preserving in data mining.
Email: nnduc@fit.hcmus.edu.vn

Bac Le is currently a Professor and Head of Department of Computer Science, Faculty of Information Technology, University of Science, Vietnam National University, Ho Chi Minh City, Vietnam. His main research includes artificial intelligence, soft computing, machine learning and data mining.
Email: lhbac@fit.hcmus.edu.vn

Graph Structure and Isomorphism Learning: A Survey

Tuyen Ho-Thi-Thanh[1,2,3]

[1] Faculty of Information Technology, University of Science, Ho Chi Minh City, Vietnam
[2] Vietnam National University, Ho Chi Minh City, Vietnam
[3] University of Economics, Ho Chi Minh City, Vietnam

Correspondence: Tuyen Ho Thi Thanh (tuyenhtt@ueh.edu.vn)
Communication: received 15 January 2022, revised 11 February 2022, accepted 27 February 2022
DOI: 10.32913/mic-ict-research.v2023.n1.1028

Abstract: With the great success of artificial intelligence in recent years, graph learning is gaining attention from both academia and industry [1, 2]. The power of graph data is its capacity to represent numerous complicated structures in a broad spectrum of application domains including protein networks, social networks, food webs, molecular structures, knowledge graphs, sentence dependency trees, and scene graphs of images. However, designing an effective graph learning architecture on arbitrary graphs is still an on-going research topic because of two challenges of learning complex topological structures of graphs and their nature of isomorphism. In this work, we aim to summarize and discuss the latest methods in graph learning, with special attention to two aspects of structure learning and permutation invariance learning. The survey starts by reviewing basic concepts on graph theory and graph signal processing. Next, we provide systematic categorization of graph learning methods to address two aspects above respectively. Finally, we conclude our paper with discussions and open issues in research and practice.

Keywords: *graph learning, graph structure learning, graph isomorphism, permutation invariance, graph neural networks.*

I. INTRODUCTION

Graphs are powerful data structures which can be considered as *a general language for describing and modeling complex systems* [3] in practice. Unlike many other structures such as images, texts and audios, which only represent the information of entities, graphs are able to capture both entities and their interactions via nodes and edges of graphs. Social networks are typical examples of graphs whose nodes are users and edges show the existence of the relationships between two users. As a consequence of their power, graphs not only play an important role in computer science but also are ubiquitous in related fields (e.g., physics, biology, chemistry) with a wide range of applications: classifying proteins in biological interaction networks, predicting the influence of an individual in social networks and discovering new drug molecules.

Basically, a graph consists of two information types: a feature matrix, denoted by $\boldsymbol{X}$, encoding entities and their properties; and a structural matrix $\mathcal{A}$ (such as adjacency matrix $\boldsymbol{A}$, Laplacian matrix $\boldsymbol{L}$ and its variants), describing the relationship between entities. Between such two types, the structural information is more crucial since object-object interaction representation is a special characteristic of graph data and different graphs are distinguishable via their topological configurations. As a result, learning graph structure becomes a central problem in graph learning. However, converting complicated raw structure information of graphs into compact fixed-size vectors in a continuous embedding space is non-trivial because of two challenges: 1) graph sizes vary dramatically in practice from several units and interactions in molecular networks to millions of objects and connections in social networks, hindering the invention of effective algorithms to extract compact representations but not losing important information; and 2) graph isomorphism, where two graphs look different but are actually the same (Fig 1), causing the requirement of permutation invariance in designing graph learning frameworks. The success of handling these challenges is the key to proceed graph learning field.

To tackle these challenges, tremendous approaches have been proposed in graph learning, ranging from graph kernel methods to graph neural networks. In this survey, we provide an overview of recent advanced learning techniques on graphs in a comprehensive manner. Our focus is effective methods that can either learn graph structures or deal with graph isomorphism.

There exists several previous comprehensive reviews related to our survey. [5] and [6] provide a thorough overview of *geometric deep learning*, a broader area, which attempts

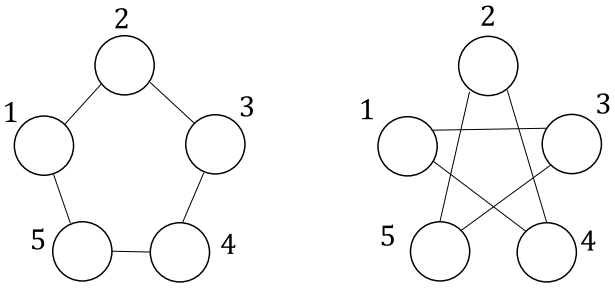


Figure 1. Two identical graphs with different looking structures (adopted from [4]).

to extend machine learning techniques for non-Euclidean data including graphs and manifolds. The early work encoding the relationships of *nodes as embedding vectors* in continuous space refers to node/graph embedding methods with detailed summary in [7, 8]. For *graph learning*, some notable surveys are [1, 9]. Mital Kinderkhedia [9] reviews over existing graph learning methods and presents them into kernel methods and graph neural networks. Meanwhile, Feng Xia et al. [1] systematically categorize the field into four approaches of graph signal processing, matrix factorization, random walks and graph neural networks. With the success of deep learning in images, there have been many methods employing deep learning techniques on graphs (aka *graph neural networks*) in recent years and the most up-to-date surveys [2, 10–12] mainly focus on this subfield of graph learning. Zhou Jie et al. [10] introduce the design pipeline of graph neural networks and provide a review on different implementations for each module in the pipeline. [2] and [11] are based on network architectures to divide graph neural network methods into groups of graph recurrent neural networks, graph convolution networks, graph autoencoders, graph reinforcement learning and graph adversarial networks. There have also been several surveys paying attention to other subfields or specific topics of graph learning such as *kernel methods* [13, 14] or constructing a unified framework for *graph convolutional networks* [12]. Although, many reviews on graph learning have been emerged, they mainly provide *technical views* with taxonomies of algorithm designs and training strategies rather than focusing on the *problem-driven aspect* as our work, i.e., solving two key problems of learning the topology and isomorphism properties of graphs. To the best of our knowledge, there is little effort to systematically summarize the field in these directions. One survey closely related our work is [15] but it only examines a limited number of works and the paper is equivalent to our Subsecs. III.2 and III.3.

In this paper, we introduce a different overview of recent advancements towards solving essential problems in graph learning. To summarize, our contributions are as follows:

- We introduce two new taxonomies of graph learning, corresponding to graph structure and permutation invariance learning. For the *graph structure learning*, existing methods are categorized into six main approaches of *graph coloring-based methods, matrix factorization, node embedding, aggregation operators* and *motif-based methods* while, to solve the latter of *graph isomorphism*, recent works are focusing on four directions of *aggregation function, ordering, histogram and permutation sampling*.
- Comprehensive survey: we provide a detailed review on numerous methods in the literature with descriptions and discussions on their advantages and disadvantages.
- Finally, we discuss on the limitations of existing methods and propose open problems for future research.

The rest of the paper is structured as follows. Sec. II presents notations, basic definitions and concepts in graph learning research, brief introduction of graph signal processing, which is the backbone of spectral graph learning methods, and Weisfeiler-Lehman test, a baseline for graph isomorphism test. We start to introduce graph structure learning and its taxonomy in Sec. III. Sec. IV outlines the approach in graph isomorphism learning and systematically summarizes different existing methods. Finally, we conclude the survey with a discussion on open challenges and potential future directions in graph learning in Sec. V and VI.

II. BACKGROUND

1. Notations

In this paper, we use bold uppercase characters to denote matrices and bold lowercase ones for vectors. For an arbitrary matrix $\boldsymbol{M}$, we use the notations of $\boldsymbol{M}(i,j)$, $\boldsymbol{M}(i,\cdot)$ and $\boldsymbol{M}(\cdot,j)$ to present a matrix element, the i^{th} row and the j^{th} column of $\boldsymbol{M}$ respectively. Table I lists our commonly used notations. Unless particularly specified, all notations are following this table. Table I also shows the abbreviations of methods mentioned in the next sections.

2. Graph

An undirected graph $G = (V, E)$ of N_v vertices/nodes and N_e edges/links consists of two sets: a set of vertices $V = \{i | 1 \leq i \leq N_v\}$ and a set of edges $E = \left\{e_{i,j} = (i,j) \,|1 \leq i, j \leq N_v\right\}$ [11]. Each vertex is associated with a N_d-dimensional feature vector $\boldsymbol{x}_i \in \mathbb{R}^{1\times N_d}$ and all vertices form a feature matrix $\boldsymbol{X} = \left[\boldsymbol{x}_1, \boldsymbol{x}_2, .., \boldsymbol{x}_{N_v}\right]^{\mathsf{T}}$. Basically, a graph G can be described by two components: a feature matrix $\boldsymbol{X}$ and a graph structure matrix $\boldsymbol{\mathcal{A}}$. Between them, graph structure is most important but difficult to obtain an effective representation form. In the following, we

Notations	Descriptions	Abbreviations	Descriptions
$\boldsymbol{G}$	Graph	1-WL	1-dimensional Weisfeiler-Lehman
$\boldsymbol{V}$	The set of nodes	CNN	Convolutional Neural Network
$\boldsymbol{E}$	The set of edges	LSTM	Long-Sort Term Memory
$\boldsymbol{A}$	Adjacency matrix	MLP	Multi-layer Perceptron
$\boldsymbol{D}$	Degree matrix	GAE	Graph Auto-encoder
$\boldsymbol{L}$	Laplacian matrix	GAT	Graph Attention Network
$\tilde{\boldsymbol{L}}$	Normalized Laplacian matrix	GCN	Graph Convolutional Network
$\boldsymbol{H}$	Hidden layer	GNN	Graph Neural Network
N_v	The number of nodes	GraphSAGE	Graph SAmple and aggreGatE
N_e	The number of edges	NMF	Non-negative Matrix Factorization
N_l	The number of network layers	RNN	Recurrent Neural Network
$\boldsymbol{X}$	Feature matrix		
$\mathcal{A}$	Structural matrix		
$\mathcal{N}(\cdot)$	Neighborhood of a node		
$\boldsymbol{s}$	Graph signal vector		
$\boldsymbol{P}$	Permutation matrix		
$\mathbb{R}_{\geq 0}$	Non-negative real numbers		
$\|\cdot\|_F$	Frobenius norm		
$\simeq$	Isomorphism		

Table I
NOTATIONS AND ABBREVIATIONS

summarize some typical structural matrices for undirected graphs in literature.

- **Adjacency matrix**: An adjacency matrix $\boldsymbol{A} \in \mathbb{R}^{N_v \times N_v}$ of an undirected graph is a symmetric matrix whose element at the i^{th} row and j^{th} column, denoted $\boldsymbol{A}(i,j)$, is equal to 1 if there exists an edge between two vertices i and j, otherwise $\boldsymbol{A}(i,j) = 0$.

$$\boldsymbol{A}(i,j) = \begin{cases} 1 & \text{if } (i,j) \in E \\ 0 & \text{otherwise} \end{cases}$$

- **Degree matrix:** A degree matrix $\boldsymbol{D} = \text{diag}\left(d_1, d_2, .., d_{N_v}\right) \in \mathbb{R}^{N_v \times N_v}$ is a diagonal matrix whose the i^{th} element on the diagonal $d_i = |\{j|\,(i,j) \in E\}|$ is the degree at the vertex i or the number of edges containing that vertex.

$$\boldsymbol{D}(i,j) = \begin{cases} d_i & \text{if } i = j \\ 0 & \text{if } i \neq j \end{cases}$$

- **Laplacian matrix**: This matrix is defined as $\boldsymbol{L} = \boldsymbol{D} - \boldsymbol{A}$. It is symmetric and diagonal and its elements are from degree matrix whilst off-diagonal elements are -1 if connected.

$$\boldsymbol{L}(i,j) = \begin{cases} -1 & \text{if } (i,j) \in E \\ d_i & \text{if } i = j \\ 0 & \text{otherwise} \end{cases}$$

- **Normalized Laplacian matrix**: The matrix $\boldsymbol{L}$ can be normalized into $[-1, 1]$ by dividing its elements by corresponding geometric mean as:

$$\tilde{\boldsymbol{L}}(i,j) = \begin{cases} -1/\sqrt{d_i d_j} & \text{if } (i,j) \in E \text{ vÃ } i \neq j \\ 1 & \text{if } i = j \\ 0 & \text{otherwise} \end{cases}$$

or rewrite it in a matrix multiplication form:

$$\begin{aligned} \tilde{\boldsymbol{L}} &= \boldsymbol{D}^{-1/2}\boldsymbol{L}\boldsymbol{D}^{-1/2} \\ &= \boldsymbol{D}^{-1/2}(\boldsymbol{D} - \boldsymbol{A})\boldsymbol{D}^{-1/2} = \mathbf{I} - \boldsymbol{D}^{-1/2}\boldsymbol{A}\boldsymbol{D}^{-1/2} \end{aligned}$$

- **Positive point-wise mutual information (PPMI)** PPMI is a probabilistic co-occurrence matrix, $\mathcal{A}_{\text{P}} \in \mathbb{R}^{N_v \times N_v}$ originated from information theory. In graph learning, PPMI is applied to encode the semantic information of entire graphs [16–18] and it is estimated via two steps: 1) compute the frequency of co-occurrence of nodes over graphs and store these values in a frequency matrix $\boldsymbol{F}$, e.g., an entry $\boldsymbol{F}(i,j)$ counts the number of random walks between two nodes i and j within a predefined window [16]; and then 2) estimate the probability $\mathcal{A}_{\text{P}}(i,j)$ that the node i occurs within a window around the node j:

$$\mathcal{A}_{\text{P}}(i,j) = \max\left(\log\left(\frac{\boldsymbol{F}(i,j)}{\left(\sum_i \boldsymbol{F}(i,j)\right)\left(\sum_j \boldsymbol{F}(i,j)\right)}\right), 0\right)$$

In addition to describe the graph structural information (vertices and their connections), the aforementioned matrices (except for PPMI matrix) can be used as graph operators. For example, multiplying an adjacency matrix $\boldsymbol{A}$ by a graph signal vector $\boldsymbol{s} = \left[s_1, s_2, .., s_{N_v}\right] \in \mathbb{R}^{1 \times N_v}$, where s_i is an observed signal/feature at the i^{th} vertex, results in a new signal $\boldsymbol{A}\boldsymbol{s} \in \mathbb{R}^{1 \times N_v}$ whose

entry is the sum of all signals of vertices j connecting to i. Meanwhile, $\boldsymbol{Ds} \in \mathbb{R}^{1\times N_v}$ is multiplying s_i by the corresponding degree value d_i at i and $\boldsymbol{Ls} \in \mathbb{R}^{1\times N_v}$ (or normalized version $\tilde{\boldsymbol{L}}\boldsymbol{s} \in \mathbb{R}^{1\times N_v}$) expresses the sum of difference between signal at i and its neighbor's signals.

3. Graph signal processing

From the perspective of graph signal processing, $\boldsymbol{A}$, $\boldsymbol{D}$,$\boldsymbol{L}$ and $\tilde{\boldsymbol{L}}$ are considered as graph representations in spatial domain. Another way of graph analysis is to convert them into frequency domain via graph Fourier transforms. The transformed graph signals show some interesting and meaningful information, e.g., connection strength between connected components, which is not obviously exposed in spatial domain, and therefore, graph signal processing is a powerful tool to learn the representation of graphs. This part provides the basic concepts in graph signal processing as an underlying theory for the approach of spectral graph learning methods.

Spectral graph analysis: Given a graph G and its normalized Laplacian matrix $\tilde{\boldsymbol{L}}$, spectral graph analysis [19] finds the spectral characteristics of G by eigen-decomposing $\tilde{\boldsymbol{L}}$ into eigen-values, described by a diagonal matrix $\boldsymbol{\Lambda}$, and a matrix $\boldsymbol{U}$ of eigen-vectors. These matrices own some important properties and therefore are considered as indicators of graph topology, for example, the number of zero eigen-values is equal to the number of connected components of G.

Graph Fourier transform: Fourier transform implies to analyze an input signal/voice/image/graph as a combination of basic components, e.g., wavelet, graphlet. Similarly, Fourier transform on graph converts a signal vector $\boldsymbol{s}$ into spectral domain encoded by eigen-vectors $\boldsymbol{U}$ of $\tilde{\boldsymbol{L}}$. Particularly, if $\tilde{\boldsymbol{L}} = \boldsymbol{U\Lambda U}^\mathsf{T}$ then Fourier transform of $\boldsymbol{s}$ is $\tilde{\boldsymbol{s}} = \mathcal{F}(\boldsymbol{s}) = \boldsymbol{U}^\mathsf{T}\boldsymbol{s}$, where $\tilde{\boldsymbol{s}}$ is the spectral representation of the input signal $\boldsymbol{s}$. The inverse graph Fourier transform projecting $\tilde{\boldsymbol{s}}$ back to the input space is given as: $\tilde{\tilde{\boldsymbol{s}}} = \mathcal{F}^{-1}(\tilde{\boldsymbol{s}}) = \boldsymbol{U}\tilde{\boldsymbol{s}} = \boldsymbol{s}$.

Graph convolution: A convolution between a signal vector $\boldsymbol{s}$ and a filter $\boldsymbol{g} \in \mathbb{R}^{N_v}$ is defined as:

$$\begin{aligned} \boldsymbol{s}*\boldsymbol{g} &= \mathcal{F}^{-1}(\mathcal{F}(\boldsymbol{s}) \odot \mathcal{F}(\boldsymbol{g})) \\ &= \boldsymbol{U}\left(\boldsymbol{U}^\mathsf{T}\boldsymbol{s} \odot \boldsymbol{U}^\mathsf{T}\boldsymbol{g}\right) = \boldsymbol{U}\left(\boldsymbol{U}^\mathsf{T}\boldsymbol{g} \odot \boldsymbol{U}^\mathsf{T}\boldsymbol{s}\right) \end{aligned}$$

where $*$ is convolution operator and $\odot$ is Hadamard product. Let $\boldsymbol{g}_\theta$ be diag$\left(\boldsymbol{U}^\mathsf{T}\boldsymbol{g}\right)$, resulting in $\boldsymbol{U}^\mathsf{T}\boldsymbol{g} \odot \boldsymbol{U}^\mathsf{T}\boldsymbol{s} = \text{diag}\left(\boldsymbol{U}^\mathsf{T}\boldsymbol{g}\right) \odot \boldsymbol{U}^\mathsf{T}\boldsymbol{s}$ and the convolution can be simplified as $\boldsymbol{s}*\boldsymbol{g}_\theta = \boldsymbol{U}\boldsymbol{g}_\theta\boldsymbol{U}^\mathsf{T}\boldsymbol{s}$. Basically, spectral graph learning methods can be distinguished based on different filters $\boldsymbol{g}_\theta$.

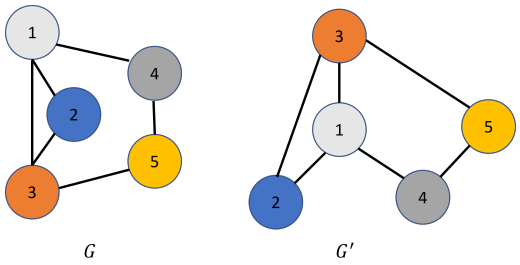


Figure 2. An example of two isomorphic graphs. The corresponding vertices share the same color (adopted from [21]).

4. Graph isomorphism

Two graphs $G = (V, E)$ and $G' = (V', E')$ are *isomorphic* $G \simeq G'$ if and only if there exists a bijection preserving the adjacency relationship of vertices. Formally, a mapping $f : V \to V'$ satisfies $\forall (u, v) \in E$ if and only if $(f(u), f(v)) \in E'$. The mapping f is called *isomorphism*. If G and $G'$are identical then $f : V \to V$ are a bijection from V to V and f is an automorphism. Fig. 2 illustrates two isomorphic graphs. Verifying whether two graphs are isomorphic is still an open problem in computer science. The recent results show that this is NP but has not been known to be NP-complete or be solved in polynomial time [20].

One of the most widely-used and effective algorithm to test isomorphism in polynomial time is Weisfeiler-Lehman test. Its most well-known is the 1-dimensional version (1-WL) [22]. Suppose that G and G' are two graphs with the size of N_v, 1-WL algorithm employs an iterative procedure (up to N_v iterations). For each iteration, 1-WL goes through all vertices of two graphs and reassigns their labels/refines their colors based on their current labels/colors and ones of their neighbors.

Let $c_{i,j}$ be the color of the j^{th} vertex of G at the i^{th} iteration and $\mathcal{N}_G(j)$ is the neighborhood of the node j of G. Colors of vertices can be sortable integers. Two nodes with the same/different color value indicate the same/different local structures. We use a multi-set $S_{i,j}$ to describe colors of that node and its neighbors at the i^{th} iteration. A multi-set is an extension of set but accepts replicated elements. 1-WL algorithm is summarized in Alg. 1.

1) Set color value to be 1 for all vertices of G and G' (lines 1-3).
2) For each iteration, enumerate all vertices of two graphs and update their colors (line 5).
   a) At each node j, gather all node's colors in its neighborhood, form a multi-set $S_{i,j}$ and sort them in ascending order (lines 6-7).
   b) Concatenate the color of the node j and $S_{i,j}$ to produce a tuple $(j, S_{i,j})$ and label this tuple with

Algorithm 1: 1-WL

1 **Inputs**: $G = \{V, E\}, G' = \{V', E'\}, |V| = |V'|, N_{\text{WL}}$
2 **Outputs**: True/False
3 **begin**
4 **for** $m \leftarrow 1, \ldots, N_v$ **do**
5 $c_{0,i} \leftarrow 1$;
6 $c'_{0,i} \leftarrow 1$;
7 **end**
8 **for** $i \leftarrow 1, \ldots, N_{WL}$ **do**
9 **for** $j \leftarrow 1, \ldots, N_v$ **do**
10 $S_{i,j} \leftarrow \text{sort}\left(\left\{c_{i-1,u} | u \in \mathcal{N}_G(j)\right\}\right)$;
11 $S'_{i,j} \leftarrow \text{sort}\left(\left\{c'_{i-1,u} | u \in \mathcal{N}_{G'}(j)\right\}\right)$;
12 $c_{i,j} \leftarrow \text{hash}\left(c_{i-1,j}, S_{i,j}\right)$;
13 $c'_{i,j} \leftarrow \text{hash}\left(c'_{i-1,j}, S'_{i,j}\right)$;
14 **end**
15 **end**
16 **if** *partitions of* G *and* G' *are the same over iterations* **then**
17 return True;
18 **else**
19 return False;
20 **end**
21 **end**

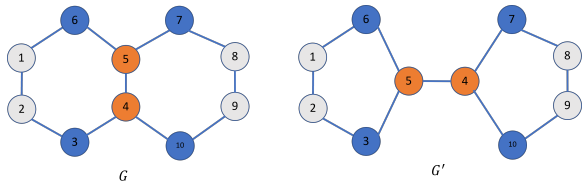


Figure 3. Non-isomorphic graphs with the same 1-WL test result (adopted from [21]).

a new color $c_{i,j}$ so that two nodes with the same tuple have the same color value (lines 8-9).

3) At the i^{th} iteration, if the algorithm divides two graphs G and G' into different partitions then two graphs are non-isomorphic and we can stop the algorithm. Otherwise, if all N_v iterations are passed and the partitions of two graphs are consistent over iterations, two graphs have high probability to be isomorphic (lines 10-13).

Basically, 1-WL produces two different colors for two graphs, they are certainly non-isomorphic. In the case of the same coloring results, we are unable to conclude there is an isomorphism between two graphs because there still exists some non-isomorphic graphs with the same 1-WL test, e.g., Fig. 3. However, WL is still powerful enough to distinguish almost all pairs of non-isomorphic graphs [23]. Fig. 4 demonstrates typical steps of 1-WL. The algorithm ends at the 3^{rd} iterations because of no changes in colors of two graphs. The complexity of 1-WL is $O\left(N_{\text{WL}} N_v\right)$, which is proportional to graph size and the number of iterations.

1-WL is a crucial algorithm in graph learning. It not only deals with the problem of testing the isomorphism of two graphs but also has been a fundamental framework to develop most models invariant to isomorphism recently.

III. GRAPH STRUCTURE LEARNING

Learning graph topology is a core topic in graph learning, whose goal is to encode entire graph structures or nodes' k-hop connections into a compact vector, which can be processed efficiently and effectively by machine learning algorithms in downstream tasks. Studies on graph structure learning are fairly diverse and their distinction lies in which part of structural information is captured and how to extract it. Overall, graph structure learning can be divided into six approaches of graph coloring, matrix factorization, node embedding, structure encoding as aggregation operator, spectral domain-based methods and motif-based methods.

1. Graph coloring based methods

Graph coloring or graph labeling is a function from a vertex set V to an ordered set (e.g., integers, real numbers, strings) and assigns a value, named color, of this target set to each node. Two nodes with similar local structures (based on predefined criteria) share the same color and vice versa, nodes with different neighborhood topologies are labeled with different colors. As a result, a color value becomes a compact embedding code describing a node's local structure.

PATCHY-SAN [24] introduces a method to extract local regions. First, based on the result of graph coloring (e.g., Weisfeiler-Lehman test [22]), the method produces a sorted list of nodes using their colors and then traverses through this list with a given stride to obtain a node sequence. Next, a breadth-first search procedure is performed to gather the neighbors of each node v with the increasing distance from v until reaching the expected number of K neighboring nodes. The graph is normalized using an extension of coloring algorithms for similar graphs, which is learned to find an optimal labeling from a collection of graphs.

[25] extracts tensorial representations of graphs using a coloring procedure twice across two dimensions. The first coloring process is performed to select a sequence of N_{sel} nodes, similar to [24]. Then, for each node, K closest neighbors are gathered and sorted based on the colors provided by another coloring procedure. Finally, we obtain a final tensor of shape $N_{\text{sel}} \times K \times N_d$, where N_d is the length of categorical one-hot vectors of nodes.

Graph coloring-based methods allow to represent complex topology of graphs into simple color codes, which are convenient for both processing and storage but still distinguishable. Moreover, the codes are designed to be orderable, rendering no requirement to explicitly store in representations, e.g., codes can be implied as tensor indices

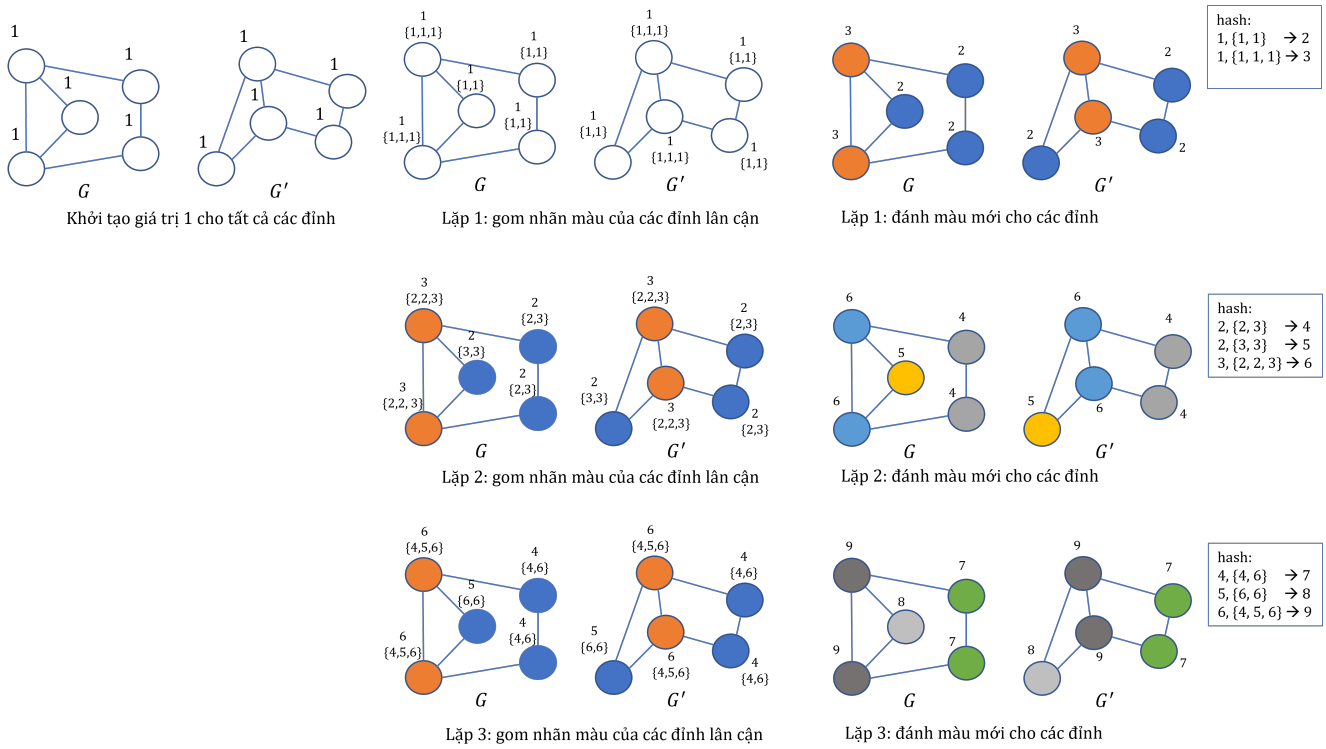


Figure 4. 1-WL procedure (adopted from [21]).

in [24, 25]. Since this structure encoding approach is completely based on graph coloring algorithms, its representation power mainly depends on the accuracy and speed of the chosen algorithms.

2. Structural matrix factorization

The topology of graphs is initially described in structural matrices such as adjacency matrix or Laplacian matrix. However, such matrices usually only provide basic information, i.e., two vertices are connected or not. For specific applications, matrix factorization can be employed to discover hidden factors by factorizing these matrices into the product of compact matrices. Resulting elementary matrices are problem-driven and help more to improve the performance of systems than the original structural matrices.

GNMTF (Graph regularized non-negative matrix tri-factorization) [26] uses non-negative matrix factorization to extract the intrinsic geometric information of networks for overlapping community detection. Particularly, given the adjacency matrix $\boldsymbol{A}$ of a graph, GNMTF aims to tri-factorize $\boldsymbol{A} \approx \boldsymbol{M}\boldsymbol{\Lambda}\boldsymbol{M}^{\mathsf{T}}$, where $\boldsymbol{M}(i,j)$ indicates the strength of the i^{th} node belonging to the j^{th} community, while the matrix $\boldsymbol{\Lambda}$ represents the relationship among communities. To find $\boldsymbol{M}$ and $\boldsymbol{\Lambda}$, GNMTF minimizes the square loss function: $\left\|\boldsymbol{A} - \boldsymbol{M}\boldsymbol{\Lambda}\boldsymbol{M}^{\mathsf{T}}\right\|_F^2$ or generalized KL-divergence $\sum\left[\boldsymbol{A}(i,j)\log\frac{\boldsymbol{A}(i,j)}{(\boldsymbol{M}\boldsymbol{\Lambda}\boldsymbol{M}^{\mathsf{T}})(i,j)} - \boldsymbol{A}(i,j) + \left(\boldsymbol{M}\boldsymbol{\Lambda}\boldsymbol{M}^{\mathsf{T}}\right)(i,j)\right]$.

ESNMF (Ego-Splitting networks using symmetric Non-negative Matrix Factorization) [27] follows the similar idea of optimization-based factorization for overlapping community detection. Instead of factorizing the adjacency matrix of the large-scale graph, ESNMF partitions it into connected subgraphs via an ego-splitting process and incorporates prior information to obtain a subgraph matrix $\hat{\boldsymbol{A}}_{\text{sub}}$. The factorization is done on this matrix: $\left\|\hat{\boldsymbol{A}}_{\text{sub}} - \boldsymbol{M}\boldsymbol{M}^{\mathsf{T}}\right\|_F^2$, where $\boldsymbol{M}(i,j)$ indicates the i^{th} node belongs to the j^{th} community, $\boldsymbol{M}(i,j) = 1$, or not $\boldsymbol{M}(i,j) = 0$.

Gaia Ceddia et al. [28] extend Non-negative Matrix Tri-Factorization to predict the interactions between drugs and proteins. An input network contains multiple types of drugs and proteins, which can be encoded as an association matrix $R \in \mathbb{R}_{\geq 0}^{|V_1|\times|V_2|}$ between two sets of nodes V_1 and V_2, whose element $R(v_1, v_2) > 0$ if $v_1 \in V_1$ and $v_2 \in V_2$ are connected; and $R(v_1, v_2) = 0$ otherwise. In [28], after introducing an enhanced association matrix $R' \in [0,1]^{|V_1|\times|V_2|}$, where $R'(v_1, v_2)$ is related to the shortest path between v_1 and v_2, the authors propose to convert R' into three non-negative factors by minimizing the following Frobenius norm: $\left\|R' - M_1 Q M_2^{\mathsf{T}}\right\|_F^2$ using an iterative update procedure. The approximate matrix $\hat{R}' = M_1 Q M_2^{\mathsf{T}}$ is used to infer novel

associations (i.e., drug-protein-disease interactions) between elements in V_1 and V_2.

Non-negative Matrix Factorization is also adopted to learn clusterable representations of nodes in a semi-supervised manner. In the proposed Unified Semi-Supervised NMF framework (USS-NMF) [29], node representations $\boldsymbol{M}$ are learned by jointly factorizing Point-wise Mutual Information matrix, label matrix, inferred cluster assignment matrix. The factorization allows connected nodes to have similar representations, rendering the capacity of local invariance node encoding. These learned representations can be used as input for any traditional classifiers such as Logistic Regression.

The above NMF-based community detection is shallow methods, which directly map from input space to community membership space. Deep Autoencoder-like Non-negative Matrix Factorization (DANMF) [30] develops a hierarchical mapping to capture the complex and diverse structures of real-world graphs. Formally, the adjacency matrix is approximated by $\boldsymbol{A} \approx \boldsymbol{M_1 M_2} ... \boldsymbol{M}_{N_l} \boldsymbol{Q}$, wherein, $\boldsymbol{M}_l$ and $\boldsymbol{Q}$ are non-negative matrices. Each column of $\boldsymbol{Q}$ reveals the possibility that a node is a member of a community whilst the series of $\boldsymbol{M}_l$ is mapping functions between the input space and the community membership space. To model this hierarchical factorization, DANMF trains a Deep Autoencoder with parameter matrices $\boldsymbol{M_l}$ and $\boldsymbol{Q}$ to minimize the difference between the original structural matrix $\boldsymbol{A}$ and its approximation. DANMF inherits both the intrinsic representation learning of deep models and the power of NMF for community detection and hence it outperforms many shallow NMF-based methods.

Node feature representations can be extracted with low rank matrix factorization. Text-Associated DeepWalk (TADW) proposes a way to incorporate text information, encoded in matrix $\boldsymbol{X}_{\text{text}}$, into a factorization framework. Let $\boldsymbol{T}$ be a transition matrix in PageRank, TADW attempts to approximate $\left(\boldsymbol{T} + \boldsymbol{T}^2\right)/2 \approx \boldsymbol{M}_1^\top \boldsymbol{Q} \boldsymbol{X}_{\text{text}}$, wherein $\boldsymbol{M}_1$ and $\boldsymbol{Q}$ are low-rank matrices. Similar to [26, 27, 29], the approximation is casted to minimization problem $\min_{\boldsymbol{M}_1, \boldsymbol{Q}} \left\| \left(\boldsymbol{T} + \boldsymbol{T}^2\right)/2 - \boldsymbol{M}_1^\top \boldsymbol{Q} \boldsymbol{X}_{\text{text}} \right\|_F^2 + \alpha \left(\|\boldsymbol{M}_1\|_F^2 + \|\boldsymbol{Q}\|_F^2 \right)$, which can be solved by alternatively updating $\boldsymbol{M}_1$ and $\boldsymbol{Q}$. Finally, $\boldsymbol{M}_1$ and $\boldsymbol{Q} \boldsymbol{X}_{\text{text}}$ are treated as the low-dimensional representations of nodes.

Designed for scalability, NetSMF (large-scale network embedding as sparse matrix factorization) [31] extracts the sparse version $\hat{\boldsymbol{L}}$ of Laplacian matrix $\boldsymbol{L}$ and constructs a NetMF matrix sparsifier with the controllable number of non-zero elements. By applying randomized SVD, which is working efficiently on sparse matrices, the authors obtain three matrix factors $\boldsymbol{U}$,Λ and $\boldsymbol{V}$ from the matrix sparsifier, and use the $\boldsymbol{U} \boldsymbol{\Lambda}^{-1/2}$ as network embeddings.

In the scenario of defending adversarial attacks, adjacency matrices are perturbed and GNNs can easily be fooled to make wrong predictions. Pro-GNN [32] attempts to learn a clean adjacency matrix $\boldsymbol{S}$ from the given perturbed matrix $\boldsymbol{A}$. The idea is to find a low-rank and sparse matrix close to $\boldsymbol{A}$ by minimizing the loss $\min_S \|\boldsymbol{A} - \boldsymbol{S}\|_F^2 + \alpha \|\boldsymbol{S}\|_1 + \beta \|\boldsymbol{S}\|_*$, where $\|\cdot\|_1$and $\|\cdot\|_*$ are L_1 and nuclear norm respectively to force $\boldsymbol{S}$ to be a sparse and low rank matrix. Learning $\boldsymbol{S}$ is done simultaneously with training GNN parameters to improve node classification task. Similarly, LDS (Learning Discrete Structures) [33] is based on an optimization framework to iteratively learn the parameters $\boldsymbol{\theta}$ of a graph generative model, which can draw edges from Bernoulli random variables $\boldsymbol{A} = \text{Ber}(\boldsymbol{\theta})$, and find the optimal parameters of a GCN given generated graphs.

For graph structure learning, matrix factorization approach enables to disentangle hidden factors, which are usually not clearly observed in original structural matrices, and therefore training machine learning methods on extracted meaningful factors yields more benefits. The main drawback of factorization-based structure learning is that it requires much expert knowledge to choose proper factorization methods for target applications, limiting the applicability of this approach. In addition, matrix factorization is also an expensive operation and infeasible to work on large graphs such as social networks or collaboration networks.

3. Node embedding

Node embedding is a group of techniques to encode nodes into low-dimensional vectors, preserving local neighborhood of nodes. More specifically, the geometric distance of two embeddings in a latent space reflects their relationship in original networks. According to [34], node embedding methods share the same 2-stage framework of encoding and decoding. An encoder is a function f^{enc} mapping from a node $v \in V$ to an embedding vector $\boldsymbol{z} \in \mathbb{R}^d$ in a d-dimensional latent space: $\boldsymbol{z} = f^{\text{enc}}(v)$. Meanwhile, a decoder f^{dec} is a mapping from the latent space to the input space, which predicts the relationship of a pair of nodes: $r = f^{\text{dec}}(\boldsymbol{z}_v, \boldsymbol{z}_u)$. Given a structural matrix $\mathcal{A}$, node embedding methods attempt to optimize the following loss: $\mathcal{L} = \sum_{u,v \in V} f^{\text{dis}}\left(f^{\text{dec}}(\boldsymbol{z}_u, \boldsymbol{z}_v), \mathcal{A}(u, v)\right)$. Existing embedding methods can be distinguished via their choice of the encoder f^{enc}, the decoder f^{dec} and the distance function f^{dis}.

High-Order Proximity preserved Embedding (HOPE) [35] defines the decoding function as inner product $f^{\text{dec}}(\boldsymbol{z}_u, \boldsymbol{z}_v) = \boldsymbol{z}_u^\top \boldsymbol{z}_v$ to capture high-order proximity and the distance function as Euclidean distance $\mathcal{L} = \sum_{u,v \in V} \left\| \boldsymbol{z}_u^\top \boldsymbol{z}_v - \mathcal{A}(u, v) \right\|$, where $\mathcal{A}$ is a proximity matrix, to ensure that the embeddings of nodes correlate with their observed connections.

Unlike HOPE which tries to reconstruct the structural matrix directly, probabilistic methods such as node2vec are based on random walks to investigate graph topology and attempt to reconstruct the probabilities of random walks. In node2vec [36], the inner product of two latent representations is connected to the conditional probability of two corresponding nodes via random walks $p\left(v|f^{\text{enc}}\left(u\right)\right) = \frac{\exp\left(z_u^{\mathsf{T}} z_v\right)}{\sum_{i,j} z_i^{\mathsf{T}} z_j}$. The entire node2vec framework reconstructs the maximum probability of linked nodes from their embeddings as follows:

$$\begin{aligned}\mathcal{L} &= \sum_{u\in V} \log\left[\prod_{v\in\mathcal{N}(u)} p\left(v|f^{\text{enc}}\left(u\right)\right)\right] \\ &= \sum_{u\in V}\left[-\log\mathcal{Z}_u + \sum_{v\in\mathcal{N}(u)} z_u^{\mathsf{T}} z_v\right], \end{aligned} \tag{1}$$

wherein $\mathcal{Z}_u = \sum_{v\in V}\exp\left(z_u^{\mathsf{T}} z_v\right)$. Similarly, DeepWalk [37] also follows the same strategy of using truncated random walks to extract the structural information and learn the topology via maximizing the probability of neighborhood.

LINE (Large-scale Information Network Embedding) [38] proposes the idea of learning both the local and global structures. Each structure type is learned via two independent models to produce different embedding vectors $z_{\text{local},u}$ and $z_{\text{global},u}$. Two embeddings are concatenated to create a final representation vector for each vertex u. The local structure is captured by the first-order proximity between nodes. In other words, it is the direct connection between nodes. The decoder as a sigmoid function of inner product of two embedding vectors is defined as the joint probability of two nodes.

$$f_{\text{local}}^{\text{dec}}\left(v,u\right) = p_1\left(v,u\right) = \frac{1}{1+\exp\left(-z_{\text{local},v}^{\mathsf{T}} z_{\text{local},u}\right)}$$

Meanwhile, the global structure or the second-order proximity is determined as the common neighborhoods of two nodes. The decoder represents the conditional probability of a context u given a node v:

$$f_{\text{global}}^{\text{dec}}\left(v,u\right) = p_2\left(u|v\right) = \frac{\exp\left(z_{\text{global},v}^{\mathsf{T}} z_{\text{global},u}\right)}{\sum_{v_1\in V} z_{\text{global},v}^{\mathsf{T}} z_{\text{global},v_1}}$$

Basically, LINE maps both the local and global structures into continuous embedding vectors that represent the probabilities over vertices. The object functions are trained to minimize the KL-divergence between the joint and conditional probabilities described by two decoders and the corresponding empirical probabilities.

For heterogeneous graphs, Heterogeneous Graph Structure Learning (HGSL) [39] first defines a weighted cosine similarity function:

$$f_{\text{r}}^{\text{dec}}\left(\boldsymbol{x}_v,\boldsymbol{x}_u\right) = \frac{1}{K}\sum_{i=1}^{K}\cos\left(\boldsymbol{w}_{k,r}\odot\boldsymbol{x}_v, \boldsymbol{w}_{k,r}\odot\boldsymbol{x}_u\right)$$

where $\boldsymbol{w}_{k,r}$ are learnable parameters, and then HGSL applies it to generate the feature similarity graph $\boldsymbol{A}_r^{\text{SF}}$, which predicts the connection probability of two nodes based on their feature vectors. By propagating feature similarity graph along topological structure $\boldsymbol{A}_r$ of graph, HGSL obtains the feature propagation graphs for head nodes $\boldsymbol{A}_r^{\text{SFH}} = \boldsymbol{A}_r^{\text{SF}}\boldsymbol{A}_r$ and tail nodes $\boldsymbol{A}_r^{\text{SFT}} = \boldsymbol{A}_r\boldsymbol{A}_r^{\text{SF}}$, which represent nodes with similar features and neighbours. These matrices are combined to obtain the feature graph $\boldsymbol{A}_r^{\text{Feat}}$. The cosine similarity function above is also used on embedding vectors of metapaths to construct a semantic graph $\boldsymbol{A}_r^{\text{Sem}}$. Finally, learned structure graph is a combination of the feature graph $\boldsymbol{A}_r^{\text{Feat}}$, the semantic graph $\boldsymbol{A}_r^{\text{Sem}}$ and the original graph $\boldsymbol{A}_r$ via an attention layer. The parameters of the graph structure learning network are jointly trained with 2-layer GCN to predict node labels.

Node embedding-based methods essentially convert discrete graph data into a richer continuous embedding space but still preserve the structural information of nodes. However, these methods suffer the following disadvantages [34]: 1) no parameters sharing between encoders cause computational inefficiency because the number of parameters grows linearly with respect to the graph size and 2) with transductive nature, these methods are unable to extract the embedding vectors of new unseen nodes without retraining embeddings. These main problems hinder the further development of this direction in recent years.

4. Structure encoding as aggregation operators

Structural matrices $\mathcal{A}$ can be viewed as 1) data objects that store the structural information of graphs (which nodes connect to which nodes) and 2) operators that are used to aggregate the information within neighborhood. Aggregation-based structure learning grounds on the latter characteristic of structural matrices. Since most convolutional graph neural networks (ConvGNN), which are gaining attention from research communities recently, are following this direction to learn graph structures, we focus on ConvGNNs in this part.

Inheriting the principles of deep learning, ConvGNNs consist of many layers of neurons, which connect to those on the next layer. Eq. 2 shows fundamental transformation in a layer, which contains a matrix multiplication between input $\boldsymbol{H}^{(l-1)}$ and learnable parameters $\Theta^{(l-1)}$ of the layer $l-1$, the neighbor aggregation via a matrix multiplication

with structural matrix $\mathcal{A}$ and finally a non-linear mapping f^{act}:

$$\boldsymbol{H}^{(l)} = f^{\text{act}}\left(\mathcal{A}\boldsymbol{H}^{(l-1)}\boldsymbol{\Theta}^{(l-1)}\right) \tag{2}$$

wherein, $\boldsymbol{H}^{(0)} = \boldsymbol{X}$ and f^{act} can be any non-linear function such as sigmoid or ReLU. Generally speaking, the input signal of each layer is transformed via a linear mapping controlled by $\Theta^{(l-1)}$ before these transformed vectors are aggregated along edges. To deeply uncover the role of $\mathcal{A}$, let's rewrite Eq. 2 into the form of neuron-wise transformation:

$$\boldsymbol{h}_v^{(l)} = f^{\text{act}}\left(\sum_{u\in\mathcal{N}(v)} \boldsymbol{h}_u^{(l-1)}\boldsymbol{\Theta}^{(l-1)}\right)$$

It can be seen that the structural matrix is only used to provide the connection information to aggregate nodes within a neighborhood, rendering an underuse of graph structures.

The early work on ConvGNNs is NN4G (Neural Networks for Graphs) [40] which directly employs adjacency matrix $\mathcal{A} = \boldsymbol{A}$ to gather the hidden representations of neighbors. Unlike Eq. 2, NN4G takes the feature matrix $\boldsymbol{X}$ of graph as well as the hidden representations of all preceding layers $\boldsymbol{H}^{(i)}$ into account.

$$\boldsymbol{H}^{(l)} = f^{\text{act}}\left(\boldsymbol{X}\boldsymbol{W}^{(l-1)} + \sum_{i=1}^{l-1}\boldsymbol{A}\boldsymbol{H}^{(i)}\boldsymbol{\Theta}^{(i,l-1)}\right)$$

where, $\boldsymbol{W}^{(l-1)}$ and $\Theta^{(i,l-1)}$ are parameters corresponding to feature and hidden representations respectively.

Graph Convolutional Networks (GCN) [41] is one of seminal works on applying deep learning to graphs. In GCN, the structural matrix is chosen as a deterministic function of adjacency matrix $\mathcal{A} = \boldsymbol{D}^{-\frac{1}{2}}(\boldsymbol{A}+\mathbf{I})\boldsymbol{D}^{-\frac{1}{2}}$, wherein $\boldsymbol{A}+\mathbf{I}$ implies self-connections added and $\boldsymbol{D}^{-\frac{1}{2}}$ works as a normalizing factor.

$$\boldsymbol{H}^{(l)} = f^{\text{act}}\left(\boldsymbol{D}^{-\frac{1}{2}}(\boldsymbol{A}+\mathbf{I})\boldsymbol{D}^{-\frac{1}{2}}\boldsymbol{H}^{(l-1)}\boldsymbol{\Theta}^{(l-1)}\right) \tag{3}$$

Deep Graph Convolutional Neural Network (DGCNN) [42] is also based on adjacency matrices to incorporate structural information but performs a different way to compute the structural matrix. For particular, by setting $\tilde{\boldsymbol{A}} = \boldsymbol{A}+\mathbf{I}$ and $\tilde{\boldsymbol{D}}(i,i) = \sum_j \tilde{\boldsymbol{A}}(i,j)$, we obtain a hidden layer update equation, similar to GCN's, but different in normalization factor:

$$\boldsymbol{H}^{(l+1)} = f^{\text{act}}\left(\tilde{\boldsymbol{D}}^{-1}(\boldsymbol{A}+\mathbf{I})\boldsymbol{H}^{(l-1)}\boldsymbol{\Theta}^{(l-1)}\right)$$

Bo Jiang et al. extend GCN to semi-supervised graph learning, where the structural matrix is not provided explicitly. The core idea of Graph Learning-Convolution Network (GLCN) [43] is to construct the structural matrix $\mathcal{A}$ by leveraging node embedding techniques (Subsec. III.3). The first layer of GLCN, parameterized by a vector $\boldsymbol{w}$ that learns the similarity of two nodes v and u using the following equation:

$$\mathcal{A}(v,u) = \frac{\boldsymbol{A}(v,u)\exp\left(\text{ReLU}\left(\boldsymbol{w}^{\mathsf{T}}|x_v - x_u|\right)\right)}{\sum_{j=1}^{n}\boldsymbol{A}(v,j)\exp\left(\text{ReLU}\left(\boldsymbol{w}^{\mathsf{T}}|x_v - x_j|\right)\right)} \tag{4}$$

In some cases, when $\boldsymbol{A}$ is not available, we drop the $\boldsymbol{A}(\cdot,\cdot)$ to yield the corresponding equation without $\boldsymbol{A}$. The next layers are typical convolutional layers, similar to GCN, but work on computed structural matrix $\mathcal{A}$ and its corresponding degree matrix $\boldsymbol{D}_{\mathcal{A}} = \text{diag}(d_1, d_2, ..., d_{N_v})$,where $d_i = \sum_{j=1}^{N_v}\mathcal{A}(i,j)$:

$$\boldsymbol{H}^{(l)} = f^{\text{act}}\left(\boldsymbol{D}_{\mathcal{A}}{}^{-\frac{1}{2}}(\mathcal{A}+\mathbf{I})\boldsymbol{D}_{\mathcal{A}}^{-\frac{1}{2}}\boldsymbol{H}^{(l-1)}\boldsymbol{\Theta}^{(l-1)}\right)$$

Finally, a perceptron layer is stacked on top to predict the label of the input graph.

The aggregation above can be viewed as a message passing process which propagates the message from a neighbor u to a node v. GAT (Graph Attention Networks) [44] assumes that neighbors send messages with different contributions to the central node v and therefore GAT introduces an attention-based mechanism to re-compute the weights of messages. The hidden state of a neuron is updated as follows:

$$\boldsymbol{h}_v^{(l)} = \sigma\left(\sum_{u\in\mathcal{N}(v)\cup\{v\}}\alpha_{vu}^{(l)}\boldsymbol{\Theta}^{(l)}\boldsymbol{h}_u^{(l-1)}\right)$$

The attention weight $\alpha_{vu}^{(l)}$ shows the contribution of a node u to a node v and is updated as $\alpha_{vu}^{(l)} = \text{softmax}\left(\text{LeakyReLU}\left(\boldsymbol{w}^{\mathsf{T}}\left[\boldsymbol{\Theta}^{(l)}\boldsymbol{h}_v^{(l-1)},\boldsymbol{\Theta}^{(l)}\boldsymbol{h}_u^{(l-1)}\right]\right)\right)$, where $\boldsymbol{w}$ and $\boldsymbol{\Theta}$ are learnable parameters.

Information aggregation using structural matrices is widely implemented in a lot of state-of-the-art graph neural network systems nowadays. But richer and higher-level structural information encoded in $\mathcal{A}$ has not been discovered yet except for some basic connections within k-hops neighborhood. There is a question about how well graph neural networks can preserve graph structures [45]. This is confirmed in [46], whose experiments show that structure-agnostic methods outperform graph neural networks on structure-driven problems (i.e., chemical datasets). Hence, more studies should be conducted to explore and extend the capacity of aggregating graph structural information.

5. Spectral domain-based methods

Spectral graph learning consists of methods that are based on graph Fourier transform and have a strong connection to the theory of graph signal processing [47–49]. Given an undirected graph whose structure can be represented by a

structural matrix $\mathcal{A}$ (adjacent matrix or Laplacian matrix). Since $\mathcal{A}$ owns special properties such as real symmetric positive semi-definite, it can be eigen-decomposed into $\mathcal{A} = \boldsymbol{U}\boldsymbol{\Lambda}\boldsymbol{U}^{\mathsf{T}}$, wherein $\boldsymbol{U} = \left[\boldsymbol{u}_1, \boldsymbol{u}_2, ..., \boldsymbol{u}_{N_v}\right] \in \mathbb{R}^{N_v \times N_v}$ is orthonormal eigen-vectors ($\boldsymbol{U}^{\mathsf{T}}\boldsymbol{U} = \mathbf{I}$) and $\boldsymbol{\Lambda}(i,i) = \lambda_i$ are eigen-values. A graph signal is defined as a vector $\boldsymbol{s} \in \mathbb{R}^{N_v}$, whose i^{th} element is an observed signal at the i^{th} vertex. Graph Fourier transform applied to $\boldsymbol{s}$ is demonstrated as $\tilde{\boldsymbol{s}} = \mathcal{F}(\boldsymbol{s}) = \boldsymbol{U}^{\mathsf{T}}\boldsymbol{s}$ and its inverse transform is $\mathcal{F}^{-1}(\tilde{\boldsymbol{s}}) = \boldsymbol{U}\tilde{\boldsymbol{s}}$. Basically, graph Fourier transform maps an observed signal in an input space to $\tilde{\boldsymbol{s}}$ in a spectral space, which is a space of columns of $\boldsymbol{U}$. In other words, the i^{th} element of $\tilde{\boldsymbol{s}}$ is a new coordinate of $\boldsymbol{s}$ in the spectral space or $\boldsymbol{s} = \sum_i \tilde{\boldsymbol{s}}_i \boldsymbol{u}_i$. A convolution between a graph signal $\boldsymbol{s}$ and a filter $\boldsymbol{g} \in \mathbb{R}^{N_v}$ is defined as:

$$\boldsymbol{s} * \boldsymbol{g} = \mathcal{F}^{-1}\left(\mathcal{F}(\boldsymbol{s}) \odot \mathcal{F}(\boldsymbol{g})\right) = \boldsymbol{U}\left(\boldsymbol{U}^{\mathsf{T}}\boldsymbol{s} \odot \boldsymbol{U}^{\mathsf{T}}\boldsymbol{g}\right)$$

where $\odot$ is Hadamard product and $*$ is convolution operator. If $\boldsymbol{g}_\theta = \text{diag}\left(\boldsymbol{U}^{\mathsf{T}}\boldsymbol{g}\right)$ then we can simplify $\boldsymbol{s} * \boldsymbol{g}_\theta = \boldsymbol{U}\boldsymbol{g}_\theta\boldsymbol{U}^{\mathsf{T}}\boldsymbol{s}$.

As a branch of ConvGNNs, spectral methods are strongly related to spatial-based ones or aggregation-based structure learning mentioned in Subsec. III.4. The basic idea is to convert the structural matrix $\mathcal{A}$ into a spectral domain, apply a filter and project back to the spatial domain. Let us revisit Eq. 2 and view the resulting multiplication $\boldsymbol{H}^{(l-1)}\boldsymbol{\Theta}^{(l-1)}$ as multi-dimensional graph signals on a graph described by $\mathcal{A}$. If we employ a filter $\boldsymbol{g}_\theta$ in a spectral space, whose basis is $\boldsymbol{U}$, the layer-wise transformation of spectral methods can be derived as:

$$\boldsymbol{H}^{(l)} = f^{\text{act}}\left(\boldsymbol{U}\boldsymbol{g}_\theta\boldsymbol{U}^{\mathsf{T}}\boldsymbol{H}^{(l-1)}\boldsymbol{\Theta}^{(l-1)}\right) \tag{5}$$

Spectral CNN [50] is one of the first studies on this direction. The transformation is done by choosing a structural matrix as Laplacian matrix and decomposing it. The transformation equation of Spectral CNN is as follows:

$$\boldsymbol{H}^{(l)}_{:,j} = f^{\text{act}}\left(\sum_{i=1}^{N_c{}^{(l-1)}} \boldsymbol{U}\boldsymbol{\Phi}^{(l)}_{ij}\boldsymbol{U}^{\mathsf{T}}\boldsymbol{H}^{(l-1)}_{:,i}\right)\left(j = 1, 2, ..., N_c{}^{(l)}\right)$$

where $N_c{}^{(l)}$ is the number of channels at the layer l. The filter $\boldsymbol{g}_\theta = \boldsymbol{\Phi}^{(l)}_{ij}$ is a diagonal matrix, whose diagonal entries can be updated during training. Eigen-decomposition is expensive and causes $O\left(N_v^3\right)$ in computational complexity. To tackle this problem, Chebyshev Spectral CNN [51] and GCN [41] are introduced to reduce the complexity using polynomial approximation.

ChebNet (Chebyshev Spectral CNN) [51] assumes that the filter $\boldsymbol{g}_\theta$ can be approximated by Chebyshev polynomial with respect to $\boldsymbol{\Lambda}$ as $\boldsymbol{g}_\theta = \sum_{i=0}^{K-1}\theta_i T_i\left(\bar{\boldsymbol{\Lambda}}\right)$, where $\bar{\boldsymbol{\Lambda}} = 2\boldsymbol{\Lambda}/\lambda_{\max} - \mathbf{I}_{N_v} \in [-1, 1]$. The advantage of Chebyshev polynomial is that each term can be recursively computed from its preceding terms $T_i(\boldsymbol{x}) = 2\boldsymbol{x}T_{i-1}(\boldsymbol{x}) - T_{i-2}(\boldsymbol{x})$ where we define $T_0(\boldsymbol{x}) = 1$ and $T_1(\boldsymbol{x}) = \boldsymbol{x}$. As a result, the convolution between a graph signal $\boldsymbol{s}$ and a filter $\boldsymbol{g}_\theta$ is obtained as:

$$\boldsymbol{s} * \boldsymbol{g}_\theta = \boldsymbol{U}\left(\sum_{i=0}^{K-1}\theta_i T_i\left(\bar{\boldsymbol{\Lambda}}\right)\right)\boldsymbol{U}^{\mathsf{T}}\boldsymbol{s}$$

Let us denote $\bar{\boldsymbol{L}} = 2\boldsymbol{L}/\lambda_{\max} - \mathbf{I}$ which can be considered as a normalizing form of Laplacian matrix $\boldsymbol{L}$ and we have $\bar{\boldsymbol{L}} = 2\boldsymbol{U}\boldsymbol{\Lambda}\boldsymbol{U}^{\mathsf{T}}/\lambda_{\max} - \mathbf{I} = \boldsymbol{U}\bar{\boldsymbol{\Lambda}}\boldsymbol{U}^{\mathsf{T}}$ and then $T_i\left(\bar{\boldsymbol{L}}\right) = \boldsymbol{U}T_i\left(\bar{\boldsymbol{\Lambda}}\right)\boldsymbol{U}^{\mathsf{T}}$. The convolution $\boldsymbol{s} * \boldsymbol{g}_\theta$ can be rewritten as:

$$\boldsymbol{s} * \boldsymbol{g}_\theta = \sum_{i=0}^{K-1}\theta_i T_i\left(\bar{\boldsymbol{L}}\right)\boldsymbol{s}$$

The new form indicates that the convolution can be approximated by recursively computing $T_i\left(\bar{\boldsymbol{L}}\right)$ in spatial domain and therefore the graph Fourier transform is completely removed and the computational cost is $O(KN_e)$. Another approximation approach is CayleyNet [52] which replaces Chebyshev polynomial with Cayley polynomial. It is proven that ChebNet is an instance of CayleyNet.

GCN (Graph Convolutional Networks) is a seminal work proposed by Kipf et al. in [41]. As a bridge between spectral and spatial approaches in ConvGNNs, GCN inherits the strength of both. From the spectral view, GCN is the first order approximation of ChebNet with $K = 1$ and $\lambda_{\max} = 2$ and the convolution equation in spatial domain is $\boldsymbol{s} * \boldsymbol{g}_\theta = \theta_0\boldsymbol{s} - \theta_1\boldsymbol{D}_{\mathcal{A}}^{-\frac{1}{2}}\mathcal{A}\boldsymbol{D}_{\mathcal{A}}^{-\frac{1}{2}}\boldsymbol{s}$. To avoid over-fitting, GCN assumes $\theta = \theta_0 = -\theta_1$. Furthermore, it uses a normalized version of adjacency matrix, i.e., $\mathcal{A} = \boldsymbol{A} + \mathbf{I}_{N_v}$ and $\boldsymbol{D}_{\mathcal{A}}(i,i) = \sum_j \mathcal{A}(i,j)$, to stabilize training and prevent gradients from exploding or vanishing. Its convolution equation is given as:

$$\boldsymbol{s} * \boldsymbol{g}_\theta = \theta\left(\mathbf{I}_{N_v} + \boldsymbol{D}^{-\frac{1}{2}}\boldsymbol{A}\boldsymbol{D}^{-\frac{1}{2}}\right)\boldsymbol{s} = \theta\boldsymbol{D}_{\mathcal{A}}^{-\frac{1}{2}}\mathcal{A}\boldsymbol{D}_{\mathcal{A}}^{-\frac{1}{2}}\boldsymbol{s}$$

DGCN (Dual Graph Convolutional Network) [16] employs two networks with the same architectures of GCN. These networks are different in input structural matrices. The first network (named $\text{Conv}_{\boldsymbol{A}}$) is based on the normalized adjacency matrix in [41] to extract local structures around nodes. The second network takes PPMI (Positive Point-wise Mutual Information) (described in Sec. II) as input structural matrix. Two networks contain their own softmax layers on top to individually predict output classes. DGCN is trained with a loss function to minimize the inconsistency between these two outputs and labels.

Among graph structure learning approaches, spectral domain-based methods have a strong theoretical background of graph signal processing, rendering its notable capacity of explainability. Particularly, it has been proven that eigen-values and eigen-vectors are associated with graph connected components, showing that spectral domain-based

methods can learn the topology of graphs. The main drawback of these methods are high computational cost required to convert signals from spatial to spectral domain but it has not been a big problem due to excellent improvement [41, 51, 52] in computation time recently.

6. Motif-based methods

Motifs are *"patterns of interconnections that occur with significantly higher frequencies in complex networks than those in random networks"* [53]. Generally speaking, network motifs are popular subgraph structures frequently occurring in real graphs. Methods in this approach are mainly based on computing the frequency of motifs in input networks. Elementary motif counting methods are not only able to represent the structures of any graphs but also are effective to measure the structural similarity between two graphs even with different number of nodes and edges. Therefore, they are able to process a lot of real-world network types varying in size and topology.

Graphlets are one of popular motifs in graph learning which were introduced in 2004 for protein-protein interaction networks [54] and are developed in graph kernel by Shervashidze et al. [55]. Since the number of graphlets exponentially increases with respect to the number of vertices, the graphlet set is limited to small graphs whose size is ≤ 5 vertices. Let $\boldsymbol{g} = \left[g_1, g_2, ..., g_{N_{\text{graphlet}}}\right]^\top$ be a collection of graphlets and $\phi_G(\boldsymbol{g})$ be a frequency vector, whose element is the frequency of occurrence of the corresponding graphlet in G. This vector can be normalized by dividing it by the sum of frequency of all graphlets in G and we gain the representation vector $\bar{\phi}_G(\boldsymbol{g})$ of G. It can be seen that the structure of a graph is encoded as a normalized frequency vector, two graphs G and G' with the similar representation vectors have a high probability to share the same graph structure. Therefore, graphlet-based methods measure the graph similarity by defining the distance between these vectors, e.g., inner product-based distance:

$$k_{\text{graphlet}}(G, G') = \bar{\phi}_G(\boldsymbol{g})^\top \bar{\phi}_G(\boldsymbol{g}') \tag{6}$$

This similarity measure can be used as kernel function in kernel-based machine learning frameworks such as SVM [56]. Other studies adopt the same idea of graph kernel in Eq. 6 but propose to use low-cost substructures such as subtree patterns or random walks.

Most works rely on Eq. 6 which assumes the independence between substructures. This is not always true. Let us consider an example on graphlets. A graphlet with size $K+1$ can be driven from a smaller graph with size K by adding nodes or edges, showing a strong relationship between graphlets. Deep Graph Kernel [57] incorporates such relationships by introducing a positive semi-definite matrix $\boldsymbol{M}$:

$$k(\boldsymbol{x}, \boldsymbol{x'}) = \phi(\boldsymbol{x})^\top \boldsymbol{M} \phi(\boldsymbol{x'}) \tag{7}$$

This matrix encodes the dependence between substructures and can be learned automatically. In some cases, if the relationship is explicit, $\boldsymbol{M}$ can be pre-computed, e.g., $\boldsymbol{M}(i, j)$ can be Levenshtein distance (edit distance - the number of node/edge addition/deletion steps to transform a substructure g_i to another substructure g_j). Eq. 7 is a general frameworks that can be applied to a lot of substructures such graphlets, subtree pattern [58, 59] or random walks [60].

HONE (Higher-order Network representation learning) [61] also uses a motif counting strategy to capture the higher-order structural dependence by first defining a set of motifs, which essentially are graphlets or orbits (graphlet automorphisms) and then building a weighted motif adjacency matrix for each motif, whose element (i, j) counts the number of occurrences of edge $(i, j) \in E$ in that motif. The structural matrix $\mathcal{A}$, which is actually a derived matrix, such as transition matrix, (normalized) Laplacian matrix, random walk-based Laplacian matrix, of each weighted motif adjacency matrix, is computed and the local/global embedding is learned for each k-step structural matrix $\mathcal{A}^k$ using matrix factorization techniques described in Subsec. III.2

To encode the higher order local structures for node embedding learning, mGCMN (graph convolutional multi-layer networks based on motifs) [62] follows the same idea of motif counting matrix similar to [61]. This means an element (u, v) of its custom motif matrix $\mathcal{A}$ indicates the number of occurrence of the nodes u and v in the corresponding motif. But unlike [61], mGCMN accepts the case when u and v are the same node. The custom motif matrix is fed into a graph neural network with convolutional layers, MLP layers and a softmax layer for a final embedding.

Ghadeer Abuoda et al. [63] leverage motifs to form high-order topological features to predict the existence of new links. The paper focuses on small motifs of size 3, 4 and 5, whose corresponding number of connected non-isomorphic motifs are 2, 6 and 21. Each motif forms a corresponding feature and therefore there are 29 features, which allows to capture the high level of topological details of graphs. Finally, a traditional classifier is built on these extracted features to perform a link prediction task.

Motif-based methods are powerful tools to learn the higher-order structures in graphs and have a wide range of applications in various areas, especially applications require structural information such as molecular or protein structures. Comparing with other graph structure approaches, motif discovery algorithms are still time-consuming in terms

of high performance time in measuring statistics of motifs in large graphs, preventing us from employing motifs with larger size (i.e., >5 nodes). In the future, further efforts are needed to accelerate motif discovery algorithms.

IV. GRAPH ISOMORPHISM

When developing machine learning systems on graphs, one has to deal with another special characteristics of graph data: graph isomorphism. An isomorphism between two graphs G and G' is a bijection between their node sets V and V' that preserves adjacency [64]. In this case, two graphs are *isomorphic*. Graph isomorphism problem is to determine whether two given graphs are isomorphic or not. This is known to be unresolved and still an open problem in computer science. The complexity of the problem is unknown. The achievements in graph isomorphism research show that its complexity is not known to be in NP-complete or be solvable in polynomial time [20]. One of the most popular and practical algorithms to check graph isomorphism until now is Weisfeiler-Lehman test [22], which allows to efficiently eliminate non-isomorphic graphs. Since graph topologies are usually represented via popular structural matrices $\boldsymbol{\mathcal{A}}$ and their properties are encoded in feature matrices $\boldsymbol{X}$, isomorphism causes row permutation in $\boldsymbol{X}$ and row/column permutation in $\boldsymbol{\mathcal{A}}$. As a result, two isomorphic graphs look different but are actually the same and there exists a permutation matrix $\boldsymbol{P} \in \{0,1\}^{N_v \times N_v}$ transforming the structural and feature matrices of this graph into ones of the other. To develop an algorithm f handling graph isomorphism, there are two important definitions related to permutation invariance as follows:

- **Permutation invariance:** [65]: a function $f(\boldsymbol{\mathcal{A}}, \boldsymbol{X})$ is permutation invariant if the output is unchanged for any node ordering $f\left(\boldsymbol{P}\boldsymbol{\mathcal{A}}\boldsymbol{P}^{\mathsf{T}}, \boldsymbol{P}\boldsymbol{X}\right) = f(\boldsymbol{\mathcal{A}}, \boldsymbol{X})$.
- **Permutation equivariance** [65]: a function $f(\boldsymbol{\mathcal{A}}, \boldsymbol{X})$ is permutation equivariant if permuting the input or the output of the function obtains the same result $f\left(\boldsymbol{P}\boldsymbol{\mathcal{A}}\boldsymbol{P}^{\mathsf{T}}, \boldsymbol{P}\boldsymbol{X}\right) = \boldsymbol{P}f(\boldsymbol{\mathcal{A}}, \boldsymbol{X})$.

In recent years, designing permutation invariant graph learning systems has been receiving increased attention in literature. Specifically, a theoretical line of studies on permutation invariant architectures and universal approximation on graphs has been introduced for graph neural networks, starting with a seminal work of Deep Sets [66]. In [66], Manzil Zaheer et. al investigate permutation invariant and equivariant functions operating on sets. The key contribution is the theorem stating that a set function is invariant to the permutation of elements in a given set from a countable domain iff it can be decomposed into the following form:

$$f(\boldsymbol{\mathcal{A}}, \boldsymbol{X}) = f_{\rho}\left(\oplus f_{\psi}(\boldsymbol{\mathcal{A}}, \boldsymbol{X})\right) \tag{8}$$

where f_{ψ} is permutation equivariant, f_{ρ} is an arbitrary function and $\oplus$ is permutation invariant operator such as sum, min, max. Eq. 8 plays an important role because most graph neural networks with permutation invariance capacity are following this fundamental architecture. However, the constraint of countable domain in [66] hinders the invariant architecture above from effective implementation for real-world applications. [67] considers a specific case of sum-decomposition, where $\oplus$ is a summation, and extends the framework to continuity on uncountable domain for more practical usefulness.

To deeply investigate the invariance and equivariance properties of graphs, some studies [68] pay attention to linear layers that are the building-blocks of most deep networks. For graph data, [68] proves that the dimensions of all permutation invariant and equivariant linear layer spaces are 2 and 15 respectively. More importantly, these dimensions are independent of the number of vertices and therefore invariant and equivariant networks can be applied to different-sized graphs. Orthogonal basis of these spaces are computed and introduced in the paper to build linear layers with invariance and equivariance capacities.

Haggai Maron et. al. [69] examine on the capacity of neural networks, whose architecture consists of a sequence of linear equivariant layers, non-liner layers and invariant layers, similar to [66], to approximate any (continuous) invariant function. The paper considers the general cases of any subgroups of a symmetric group acting on $\mathbb{R}^n$ by permuting coordinates. Suppose that a G-invariant function is a function satisfying $f(g \cdot x) = f(x)$ for all element $x \in \mathbb{R}^n$ and action g in the subgroup. Results from the paper show that a G-invariant network can be a universal approximator of arbitrary continuous G-invariant function if high-order tensors are allowed. If group is a set, only the first order G-invariant network (with maximal tensor order of 1) is required to be universal and one instance is shown in [66]. The work of k-IGNs [69] is the first step to discover the approximation power of invariant networks in general and deep networks on graphs in particular, rendering better understanding of the limitation and capacity of invariant architecture. The work of [69] is confirmed in [70], which introduces an alternative proof based on Stone-Weierstrass theorem for algebra of real-valued functions. Furthermore, [70] extends the results to equivariant functions using a novel generalized version of Stone-Weierstrass theorem.

It is notable that not all graph learning methods are equipped with permutation invariant or equivariant operators (for example, GraphSAGE with LSTM aggregation [71]) and hence, theoretically, they are unable to handle isomorphic graphs in practice. This section reviews common techniques to train permutation invariant graph learning systems.

Further, although permutation invariance and equivariance are equally crucial in theory, invariance property has been receiving more attention for graph data due to its direct connection to graph isomorphism problem. Hence, we mainly focus on introducing graph learning methods with permutation invariance in the following part and leave the review on permutation equivariant systems for future work.

1. Aggregation function (readout function)

A fundamental approach to build a permutation invariant system relies on ordering invariant aggregation functions. Aggregation functions (or readout functions or pooling layers) are used to gather the information of a node and nodes/edges in its neighborhood. Most methods in this approach follows the framework proposed in [66] with a permutation equivariant function f_{ψ} and permutation invariance aggregation operator $\oplus$ followed by an arbitrary function f_{ρ} (Eq. 8). The most popular permutation invariant aggregators are sum, product, minimum, maximum, covariance, some of which are reviewed in this section.

K. Xu, W. Hu, el al[72] introduces a framework, named Graph Isomorphism Network (GIN), which is as powerful as Weisfeiler-Lehman graph isomorphism test. To ensure the permutation invariance capacity, a readout function/pooling layer aggregates node features to yield graph representations. The paper examines three permutation invariant functions including sum, mean and max aggregators and ranks them by expressive power. Namely, one with the most representational power is sum that describes the whole multi-set whilst mean expresses the distribution of the multi-set and max reduces the multi-set to a simple set. Since mean and max aggregators are not injective, they are weaker in distinguishing isomorphism than sum operators. This explains the common choice of sum operators in existing invariance systems [67, 72].

Set Transformer [73] is designed to model the interactions of input set elements. The proposed architecture consists of an encoder, which is equivariant to permutation, and a decoder, whose pooling layer is a permutation invariant transformation or essentially is a sum of outputs from the encoder. Unlike most popular networks, which are based on GCNs or their variants, Set Transformer is completely built on layers with attention mechanism and therefore it allows to learn the high-order connections between elements in sets.

Besides providing permutation invariance, some studies such as Bilinear GNN [74] extends the capacity of aggregators to capture interactions between neighbor nodes, which is not encoded by existing models. To that end, Bilinear GNN computes the sum of all node pairs in the neighborhood of a node v, namely $f(v) = \frac{2}{d_v(d_v+1)} \sum_{i \in \mathcal{N}(v) \cup \{v\}} \sum_{j \in \mathcal{N}(v) \cup \{v\} \& i<j} \boldsymbol{h}_i \boldsymbol{W} \odot \boldsymbol{h}_j \boldsymbol{W}$, where d_v is the degree of the node v. The interaction between two nodes is effectively modeled via element-wise product between their representation vectors. It is notable that the interaction aggregation takes the central node into account to enrich the aggregation result in the case of sparse local structures with several neighbors.

A drawback of aggregation operators of sum, max or min is its scalar output, causing the significant loss of information, especially for local structures with many nodes. Based on the idea of vector output in capsule networks, GCAPS-CNN (Graph Capsule Convolutional Neural Networks [75] is able to capture higher order statistical information of adjacent nodes. Given a hidden representation vector $\boldsymbol{H}^{(l)}$ at the layer l, GCAPS-CNN calculates the covariance of $\boldsymbol{H}^{(l)}$, which is known to be invariant to permutation, as $f\left(\boldsymbol{H}^{(l)}\right) = \frac{1}{N}\left(\boldsymbol{H}^{(l)} - \mu\right)^{\mathsf{T}}\left(\boldsymbol{H}^{(l)} - \mu\right)$, where μ is the mean vector of $\boldsymbol{H}^{(l)}$. Compared with scalar aggregators, much information of data distribution is preserved in covariance such as shape, norm and angles between node features. GCAPS-CNN can be classified as a permutation invariance aggregator based on matrix multiplication.

Similarly, PiNet (Permutation Invariant Graph Neural Network) [76] is an end-to-end permutation invariant network using matrix multiplication. A feature matrix $\boldsymbol{X}$ and an adjacency matrix $\boldsymbol{A}$ are passed through neural layers to obtain the representation matrices $\boldsymbol{Z}_{\boldsymbol{X}} \in \mathbb{R}^{N_v \times N_{\boldsymbol{X}}}$ and $\boldsymbol{Z}_{\boldsymbol{A}} \in \mathbb{R}^{N_v \times N_{\boldsymbol{A}}}$ respectively, where $N_{\boldsymbol{X}}$ and $N_{\boldsymbol{A}}$ are the corresponding hidden dimensions. Next, these two matrices are combined in a product layer $\boldsymbol{Z}_{\boldsymbol{X}}^{\mathsf{T}} \boldsymbol{Z}_{\boldsymbol{A}} \in \mathbb{R}^{N_{\boldsymbol{X}} \times N_{\boldsymbol{A}}}$, which is invariant to node orders$(\boldsymbol{P}\boldsymbol{Z}_{\boldsymbol{X}})^{\mathsf{T}}(\boldsymbol{P}\boldsymbol{Z}_{\boldsymbol{A}}) = \boldsymbol{Z}_{\boldsymbol{X}}^{\mathsf{T}}\left(\boldsymbol{P}^{\mathsf{T}}\boldsymbol{P}\right)\boldsymbol{Z}_{\boldsymbol{A}} = \boldsymbol{Z}_{\boldsymbol{X}}^{\mathsf{T}}\boldsymbol{Z}_{\boldsymbol{A}}$ for any permutation matrix $\boldsymbol{P}$. Basically, the element (i, j) of the product matrix is the inner product of the i^{th} feature-related representation vector and the j^{th} structural representation vector across all nodes.

Due to the efficiency and scalability, especially for huge social networks, aggregation-based approach is the most practical choice to develop graph learning systems with permutation invariance power. However, compressing huge amount of neighborhood information into single scalars or vectors via aggregators such as sum or max causes the significant loss of information and results in the misclassifcation between non-isomorphic graphs. Although high-order tensors [69] help to improve the power of discrimination, they add more computation cost. Therefore, developing effective aggregation methods that balance the representation power and scalability is a challenging question in this research direction.

2. Ordering-based approach

To overcome the loss of information when using aggregation functions, a few attempts have been done to preserve the information of neighbor nodes but still satisfy permutation invariance constraint by sorting them to form a consistent sequence, which is the same for any node permutation. The idea is based on the basic property of sorting: given a sequence of real-valued numbers, sorting algorithms produce the same ordered sequence, e.g., ascending or descending order, for its permuted sequences and the ordered sequence can be considered as canonical form of the input sequence. As a result, two isomorphic graphs have the same representation after sorting step and then the goal of permutation invariance is obtained.

Deep Graph Convolutional Neural Network (DGCNN) [4] proposes a sorting-based pooling layer, named SortPooling, to reorder nodes by representation values in a consistent and meaningful manner. Support that after passing all graph convolution layers, the concatenated representation matrix is $\boldsymbol{H}^{(1:N_l)} \in \mathbb{R}^{N_v \times N_c}$,where $N_c = \sum_{l=1}^{N_l} {N_c}^{(l)}$ is the total number of channels over all representation layers, $\boldsymbol{H}^{(1:N_l)}$ is sorted row-wise according to the last column of $\boldsymbol{H}^{(1:N_l)}$. If two rows have the same last column values, the second last and the next one is used for comparison until an order can be identified. Basically, this sorting strategy is colexicographical ordering. The proposed SortPooling has two key advantages compared with scalar aggregation operators: (1) much information is allowed to pass, rendering more meaningful representations and (2) by storing the order of SortPooling layer's input, the initial state of the node (before sorting) can be restored and this also enables to do backpropagation for SortPooling.

PATCHY-SAN [24] is a graph learning method that strictly follows the intuition of CNNs' filters on local receptive fields with fixed size and defined neighboring order. To that end, for a selected node v, PATCHY-SAN first extracts K neighbor nodes using a breath-first search algorithm with increasing distance from the target node v and find an optimal order for this K-node sequence. Unlike the aforementioned methods which simply perform a sorting procedure, PATCHY-SAN implements an optimization step to find an optimal solution as canonical ordered sequence. In particular, the graph labeling optimization problem is $\hat{l} = \underset{l}{\operatorname{argmin}}\, \mathbb{E}\left[\left|f_{\boldsymbol{A}}^{\mathrm{dis}}\left(\boldsymbol{A}_{G'_K}, \boldsymbol{A}_{G''_K}\right) - f^{\mathrm{dis}}\left(G'_K, G''_K\right)\right|\right]$, where $f_{\boldsymbol{A}}^{\mathrm{dis}}$ and f^{dis} are distance functions on graphs and adjacency matrices with K nodes. By solving this object function, we can obtain an optimal node order $\hat{l}$ such that, for any uniformly random graphs G'_K and G''_K from a graph collection with K nodes, the expected difference between random graphs is minimized. The advantage of optimization-based approach is that it not only works on isomorphic but also similar graphs with different connections and therefore more generalization in practice.

For molecule related problems, Coulomb matrix is adopted to describe the basic properties and structures of molecules and it is formulated as follows:

$$C(i,j) = \begin{cases} 0.5 z_i^{2.4} & \forall i = j \\ \frac{z_i z_j}{|\boldsymbol{r}_i - \boldsymbol{r}_j|} & \forall j \neq j \end{cases}$$

where, z_i and $\boldsymbol{r}_i$ are the charge and 3D position of the i^{th} atom in a given molecule. Basically, the elements on the diagonal indicate polynomial fit of the potential energies of the free atoms whilst the other elements are Coulomb repulsion between nuclei pairs. Since Coulomb matrix lacks the capacity of permutation invariance, rendering the exponential blow-up of Coulomb descriptors for the same molecule. [77] proposes to transform C to spectral form by computing the eigen-decomposition and use the sorted eigen-values of C as permutation invariant representation vector. Gregoire Montavon et. al [78] introduce another solution in spatial domain by reordering rows of C according to their norms. To deal with larger dimensionality, the paper also introduces the idea of adding a random noise to row norms before sorting to generate more sorted Coulomb matrices per molecule for data augmentation. Among three methods, randomly sorted Coulomb representation produces the best performance for the problem of atomization energy prediction [78].

EigenPool (Eigenvector-based Pooling Layers) [79] develops permutation invariance operators by sorting eigenvalues in spectral domain. EigenPool is a graph convolutional network that consists of coarse-to-fine groups of convolution layers and pooling layers. The pooling layers have two steps of: (1) spectral clustering to divide the graph at the level l into several non-overlapping clusters and then (2) nodes with the same cluster are aggregated by summing their feature vectors. Since the spectral clustering always gives the same clustering results for all node permutation, the final graph representation is permutation invariant. Spectral graph clustering first converts initial graphs in spatial domain into spectral domain by applying graph Fourier transform on graph Laplacian matrix. The top K eigen-vectors are sorted by their corresponding eigenvalues in an ascending order to form a matrix $\boldsymbol{U}$ of $N_v \times K$, whose rows represent the clustering features of the corresponding nodes in spectral space. A clustering algorithm, such as K-means, can be applied to group nodes into clusters. Since the clustering feature of each node is unchanged under permutation transformation, the clustering results are completely independent of node order.

Graph isomorphism problem has a strong connection to graph ordering that is to find an optimal order for all nodes of a graph [80]. Graph ordering can be thought as a way to convert graphs into a canonical form, which is identical for any isomorphic graphs, rendering an invariance to node permutation. [80] proposes a graph ordering neural network, called Deep Ordering Network with Reinforcement Learning (DON-RL), towards graph visualization applications, where isomorphic graphs should have the consistent and unique visualization in a space. The key intuition is to find an optimal node ordering that maximizes the following accumulated locality score $f(\pi) = \sum_{0 \leq \pi_u - \pi_v \leq w} f^{\text{sim}}(v, u)$, where u and v are two ordered nodes in a windows of size w and $f^{\text{sim}}(v, u)$ is a similarity function. To overcome the combinatorial explosion of training space, DON-RL is trained to predict the next node, which should be added to the current ordered node subset of V. The selected node should be one that maximizes the accumulated locality score. By adding each node from V to the subset until all nodes are placed, a node ordering is obtained. It can be seen that DON-RL is similar to PATCHY-SAN[24] in terms of automatically learning node ordering, but unlike [24], optimization is obtained via more advanced Reinforcement Learning framework in DON-RL.

The benefit of ordering-based approach compared with aggregation function is to allow much information pass through the pooling/sorting layers. This helps to construct more distinctive and meaningful features for downstream tasks. The main drawback of this approach is more computation cost is required to arrange sequences in an expected order. Sometimes, due to noise and approximation errors (e.g., of eigen-decomposition implementation or optimization algorithms), sorting results are not always the same as in theory. However, ordering-based approach, along with aggregation functions, are still effective solutions for building graph learning systems invariant to node permutation.

3. Histogram-based methods

Histograms are fundamental and popular representations across many fields. Basically, a histogram is a frequency vector, where each index is associated with an object of interest and the corresponding element represents the number of times that object occurs. In graph learning, objects are substructures including subgraphs (also known as graphlets, motifs or graph fragments) [81] or special structures such as shortest paths, subtrees, random walks [24], cycles or cliques [82]. Given a dictionary $\mathbf{\Sigma}$ of common substructures, the idea of histogram-based approach to encode a graph is to count the frequency of substructures in the graph and map to a fixed size and ordered representation vector. Although graph isomorphism impacts on the order/label of nodes in graphs, local structures are unchanged and hence frequency vectors are consistent for isomorphic graphs. In other words, two isomorphic graphs share the same substructure distributions. However, since this approach does not take positional relationship of substructures into account, two graphs with different local substructure arrangement still have the same representation vector.

Substructure counting techniques are common in graph kernel methods that are based on idea of designing a kernel function to measure the similarity between any two graphs and then applying kernel methods, e.g., SVMs [56], to perform machine learning tasks. An example is the early work of graphlets [55], where the kernel function is defined as inner product between two frequency vectors, which count the frequency of graphlets with size of $k \leq 5$, namely $k_{\text{graphlet}}(G, G') = \bar{\phi}_G(\boldsymbol{g})^{\mathsf{T}} \bar{\phi}_G(\boldsymbol{g}')$. Other substructures with lower computation cost can be used, for example, subtree patterns [57] or random walks [57]. Computing frequency of a graphlet with size of k requires to enumerate all graphlets of size k in a given input graph. The complexity is $O(N_v^k)$, which is expensive. Sampling is an alternative solution which draws a sufficient number of graphlets randomly from the input graph. The expectation is that if the number of drawn samples is large enough, the approximate graphlet distribution is close to the true distribution. By choosing the proper number of samples, this approach allows to balance between the accuracy and the efficiency in practical applications, especially ones related to large graphs.

Instead of building a huge graphlet dictionary from training data and suffering the expensive graph encoding cost due to too many graphlets need to be counted [57], Dimistris Floros et. al [83] use a predefined dictionary of limited size of 16 graphlets to speed up the process. For any input graph with N_v vertices, an vertex-graphlet incident matrix $\in \mathbb{R}^{N_v \times |\mathbf{\Sigma}|}$ to describe the occurrence frequency of a graphlet in $\mathbf{\Sigma}$ at the i^{th} node. As a result, any two nodes at the same orbit (a subset of vertices symmetric under permutations [82, 83]) have the same frequency vector, resulting in orbit-invariance.

With the dominance of message passing based GNNs in literature, [82] introduces GSN (Graph Structure Networks) to overcome the limitations of conventional GNNs. Although substructures play an important role in many studies of networks, e.g., molecules or proteins, existing GNNs mostly discover the features of a node itself or the aggregated features of its neighbors rather than its local structures. As the result, the expressive power of GNNs is only equivalent to Weisfeiler-Lehman test. To leverage the source of rich information from substructures, GSN proposes to integrate a subgraph isomorphism counting function

for node feature encoding. Let $\mathbf{\Sigma}$ be the collection of small substructures, e.g., cycles of fixed length or cliques. For a substructure $g^{\mathbf{\Sigma}} \in \mathbf{\Sigma}$, let $\mathrm{Aut}\left(g^{\mathbf{\Sigma}}\right)$ denote the set of all unique automorphism of $g^{\mathbf{\Sigma}}$. For any input graph G, support that $g \in G$ is a subgraph of G and isomorphic to $g^{\mathbf{\Sigma}}$ via a bijection mapping f^{iso} from V_g to $V_{g^{\mathbf{\Sigma}}}$, the structural feature at node v with respect to substructure $g^{\mathbf{\Sigma}}$ is defined as:

$$\boldsymbol{x}_v^{g^{\mathbf{\Sigma}},i} = \frac{\left|\left\{g \simeq g^{\mathbf{\Sigma}} : v \in g \text{ s.t. } f^{\mathrm{iso}}(v) \in O_{g^{\mathbf{\Sigma}},i}\right\}\right|}{\left|\mathrm{Aut}\left(g^{\mathbf{\Sigma}}\right)\right|}$$

$$\boldsymbol{x}_v^{g^{\mathbf{\Sigma}}} = \left[\boldsymbol{x}_v^{g^{\mathbf{\Sigma}},1}, \boldsymbol{x}_v^{g^{\mathbf{\Sigma}},2}, ..., \boldsymbol{x}_v^{g^{\mathbf{\Sigma}},N_{g^{\mathbf{\Sigma}}}}\right]$$

where $O_{g^{\mathbf{\Sigma}},i}$ is the i^{th} unique element of the quotient of the automorphism acting on the substructure $g^{\mathbf{\Sigma}}$. By enumerating all substructures $g_i^{\mathbf{\Sigma}}$ in $\mathbf{\Sigma}$ and concatenating their structural features, we obtain the vertex structural feature of a node v as: $\boldsymbol{x}_v = \left[\boldsymbol{x}_v^{g_1^{\mathbf{\Sigma}}}, \boldsymbol{x}_v^{g_2^{\mathbf{\Sigma}}}, ..., \boldsymbol{x}_v^{g_{|\mathbf{\Sigma}|}^{\mathbf{\Sigma}}}\right]$. Another edge-related structural feature is introduced in [82]. These structural features along with conventional hidden vertex features $\boldsymbol{h}_v^{(l)}$ are plugged into a message passing framework [84] defined in a general manner. GSNs are proven to be at least as powerful as MPNNs (Message Passing Neural Networks) [84] and 1-WL test.

Unlike nodes or edges, which are basic elements in graphs, substructures are higher-order components containing crucial topological information and unique characteristics of a graph. These components can help to distinguish different graphs better than many permutation invariance techniques [82]. But recognizing substructures of interest in a large graph is not a trivial task because substructures are essentially graphs and also affected by isomorphism. Furthermore, enumerating large graphs for higher-order substructure counting is time consuming, hindering the popularity of this approach and its applicability in practice. However, recent advances in discovering theoretical expressive power [85] of substructures and their practical representation capacity of real complex topology reveal their potential to develop more powerful graph representation systems.

4. Permutation sampling

Another line of studies, which does not explicitly design permutation invariance functions but forces instead models to deal with permutations, is permutation sampling. The intuition is to train models on data, which is randomly permuted. It seems to be unsound in theory at the first glance but it is actually associated with exchangeability in probability distributions [86, 87].

One of general form of permutation sampling is Janossy pooling [86] that represents permutation invariant functions as the average of a permutation-sensitive function acting on all reorderings of input sequences. Let $f^{\mathrm{arb}}(\cdot)$ denote arbitrary function that can accept any variable-size finite input sequences $\boldsymbol{h}$ with the length of $|\boldsymbol{h}|$, and π is a permutation in a set $\Pi_{|\boldsymbol{h}|}$ of all permutation of $\boldsymbol{h}$.

$$f(|\boldsymbol{h}|, \boldsymbol{h}) = \frac{1}{|\boldsymbol{h}|!} \sum_{\pi \in \Pi_{|\boldsymbol{h}|}} f^{\mathrm{arb}}(|\boldsymbol{h}|, \boldsymbol{h}) \tag{9}$$

Theoretically, $\Pi_{|\boldsymbol{h}|}$ and sum are invariant to the order of input sequences and hence $f(|\boldsymbol{h}|, \boldsymbol{h})$ becomes a permutation invariant function. Computing the sum over permutation set $\Pi_{|\boldsymbol{h}|}$ is expensive, an approximation is required for computational tractability. In [86], the authors propose three methods of approximation: 1) convert $\boldsymbol{h}$ into a canonical form, e.g., sorting (Subsec. IV.2), before feeding it into $f^{\mathrm{arb}}(\cdot)$; 2) still employ Eq. 9 but on some small first k elements of input sequences; and 3) evaluate the sum via some random permutation samples from $\Pi_{|\boldsymbol{h}|}$ instead of all its elements. These methods offer significant computational savings and can be implemented in practice. Relational Pooling [88] is an extension of Janossy Pooling. Although they share the same idea of summing over all permutation set, Relational Pooling works directly on the structural matrix and the feature matrix $f(G) = \frac{1}{|V|!} \sum_{\pi \in \Pi_{|V|}} f^{\mathrm{arb}}\left(\boldsymbol{\mathcal{A}}_{\pi,\pi}, \boldsymbol{X}_{\pi}\right)$. The experiments in the paper shows that RP-GNN, which is actually GNN with Relational Pooling, is more expressive than Weisfeiler-Lehman test [88]. It means that RP-GNN can distinguish pairs of non-isomorphic graphs that cannot be recognized by Weisfeiler-Lehman test.

Local Relational Pooling (LRP) [81] adopts the framework of Relational Pooling [88] but applies to local structures of egonets. An egonet of a depth r at a specific node v is an induced subgraph including v and its neighborhood $\mathcal{N}_v^r$ that consists of all nodes within the distance of r. For Relational Pooling, we have to consider all permutations of nodes in $\mathcal{N}_v^r$. However, since egonets are rooted graphs, the computational cost reduces to a subset of $\mathcal{N}_v^r$, which is equivalent to breath-first-search (BFS) at v. Formally, Local Relational Pooling is defined as: $f(v) = \frac{1}{|V^{\mathrm{BFS}}|!} \sum_{\pi \in \Pi_{|V^{\mathrm{BFS}}|}} g\left(\boldsymbol{\mathcal{A}}_{\pi,\pi}^{v}\right)$. To further improve the speed, only k ordered nodes in BFS list is considered rather than taking all nodes in the egonets. For a network with a bounded degree, the complexity of the pooling layer grows linearly with respect to the number of vertices.

Edo Cohen-Karlik et. al [89] adopt the idea of [86] by feeding randomly permuted input sequences but the authors apply it to building a regularizer towards permutation invariance for RNNs. The goal of RNNs is to train a function: $f : \mathcal{S} \times \mathcal{X} \rightarrow S$ starting from an initial state $\boldsymbol{s}_0$. The state at the time $t+1$ is updated via the previous state as: $\boldsymbol{s}_{t+1} = f(\boldsymbol{s}_t, \boldsymbol{x}_t)$. To learn a permutation invariance func-

tion f, which produces the same output regardless of any input order, [86] introduces subset invariance regularization (SIRE). For a training subset with the same output, SIRE should be zero otherwise it should penalize for different outputs. More interestingly, the authors prove that this can be obtained efficiently by swapping any two elements $\boldsymbol{x}_1$ and $\boldsymbol{x}_2$ during training, resulting in the following regularization term: $\mathbb{E}_{\boldsymbol{s}\in\mathcal{S}}\left[\left(f\left(\boldsymbol{s},\boldsymbol{x}_1,\boldsymbol{x}_2\right)-f\left(\boldsymbol{s},\boldsymbol{x}_2,\boldsymbol{x}_1\right)\right)^2\right]$, where $\boldsymbol{x}_1$ and $\boldsymbol{x}_2$ are two new inputs and $f\left(\boldsymbol{s},\boldsymbol{x}_i,\boldsymbol{x}_j\right)=f\left(f\left(\boldsymbol{s},\boldsymbol{x}_i\right),\boldsymbol{x}_j\right)$. Moreover, RNNs trained with SIRE use fewer parameters than Deep Set based approach [66].

Some studies [71, 90] follow the strategy of permutation sampling but reduce the computational cost by drawing only one sample instead of a large batch of samples. Deep Collective Inference (DCI) [90] is an example, which implements a RNN-based framework for collective inference. Namely, for a particular node, DCI transforms the target node and its neighbors into an unordered node sequence before feeding into a RNN to predict corresponding class outputs. Meanwhile, GraphSAGE [71] offers to uniformly sample the fixed number of nodes in neighborhood in its aggregation step.

Permutation sampling approach can be applied to learn permutation invariance in many complex models, such as LSTMs or RNNs, for which it is challenging to explicitly design a permutation invariance function. Additionally, sampling allows significant computational savings, which is especially helpful for training heavy models. Although this approach is still gaining less attention from community in both theoretical and practical research, recent achievements [71, 90] are fairly promising.

V. DISCUSSION

In this section, we briefly summarize and analyze existing approaches and open issues for graph structure and permutation invariance learning:

Graph structure learning: *Structural matrix factorization* and *spectral domain-based methods* are classical approaches and usually face the problem of high computational cost, hindering their applicability in real-world scenarios. *Node embedding* are node-level models without parameter sharing and therefore they are not suitable for popular applications related to entire graphs. Two existing approaches, which most graph learning studies are focusing on, are ones based on *graph coloring* and *aggregation operators*. These techniques can be observed in recent state-of-the art graph neural networks nowadays due to their efficiency and accuracy. However, they usually rely on simple and easy-to-compute substructures and therefore discovering higher and complex structures should be considered and explored. For *motif-based methods*, although they are very powerful to represent local topology, their expensive cost is limiting the number of research on this direction. But integrating motifs into existing graph learning systems will be a potential to improve the power of graph structure learning.

Permutation invariant learning: *Histogram-based methods* are widely used in kernel methods but have not been studied in recent years because of their high cost to find patterns of interest in large graphs to construct histograms. By contrast, *aggregation function* and *ordering-based approaches* are dominating the literature to date due to better balance between accuracy and speed. Compared with ordering-based approach, aggregation functions are much faster but they usually have the problem of information loss. However, recent theoretical and practical results show that aggregation functions with higher-order tensors can expand the capacity of distinguishing isomorphic graphs, shining a light onto building better permutation invariance systems. For complex graph learning systems, which are infeasible or costly to integrate permutation invariance functions by design, training permutation invariance models using *permutation sampling* is a potential direction with some promising results [71, 90]. However, since there are a few studies following this direction, further research should be conducted to comprehensively verify and evaluate its effectiveness in both theoretical and practical aspects.

Some open issues of both graph structure and isomorphism invariance learning can be listed as follows:

- **Scalability:** Designing systems to work efficiently on large graphs is not only a big challenging for graph learning but also for any graph-related problems. Basically, the number of substructures and patterns increase according to graph size, resulting in a significant increase in the cost of enumerating nodes and encoding local structures. Many graph structure learning methods depend on fundamental structure matrices, which become inefficient on such large graphs. For isomorphism test, comparing two large graphs to verify whether they are isomorphic or not is expensive as well. Therefore, there is an open question about developing better algorithms that balance between speed and accuracy for large-scale graphs.
- **Complex Graphs**: The majority of graph structure learning and permutation invariance studies are conducted on undirected homogeneous graphs. But more sophisticated graphs can be seen in real-world applications. Some examples are directed graphs that imply the direction of the relationship between entities or heterogeneous groups whose nodes and edges can have different types or dynamic graphs whose nodes and edges change over time. All such types of graphs usually require algorithms with proper designs and

optimizations in order to maintain good performance in real-world scenarios.

- **Higher-order substructures:** High-order substructures often contain more meaningful information. This not only allows to describe complicated graph topology but also helps to better distinguish more non-isomorphic graphs. However, since substructures are actually graphs, recognizing and encoding them require more cost with respect to their structure complexity. With the lack of efficient methods to process higher-order substructures, most existing graph learning models are still based on simple graph elements for efficiency. But we believe that, with the rapid growth and development of deep learning in recent years, discoveries and advances on higher-order substructures can bring us more powerful graph learning systems in the near future.

VI. CONCLUSION

This survey has given a comprehensive review on graph learning advances in recent years. Particularly, it focuses on two fundamental but important topics of graph learning, which are graph structure learning and permutation invariance learning. For each topic, we examine existing approaches and provide detailed analysis on its pros and cons. Future directions of these topics are also discussed and explained. We hope that this survey can help researchers to better understand these topics as well as the broader field of graph learning, one of hot areas in machine learning nowadays.

REFERENCES

[1] F. Xia, K. Sun, S. Yu, A. Aziz, L. Wan, S. Pan, and H. Liu, "Graph learning: A survey," *IEEE Transactions on Artificial Intelligence*, pp. 1–1, 2021.

[2] Z. Zhang, P. Cui, and W. Zhu, "Deep learning on graphs: A survey," 2018.

[3] J. Leskovec, "Representation Learning on Networks," *WWW-18 Tutorial*, 2018. [Online]. Available: http://snap.stanford.edu/proj/embeddings-www/

[4] M. Zhang, Z. Cui, M. Neumann, and Y. Chen, "An End-to-End Deep Learning Architecture for Graph Classification," in *Proceedings of the 32nd Conference on Artificial Intelligence (AAAI)*, New Orleans, Louisiana, USA, February 2-7 2018, pp. 4438–4445.

[5] W. Cao, Z. Yan, Z. He, and Z. He, "A comprehensive survey on geometric deep learning," *IEEE Access*, vol. 8, pp. 35 929–35 949, 2020.

[6] M. Bronstein, J. Bruna, Y. Lecun, A. Szlam, and P. Vandergheynst, "Geometric deep learning: Going beyond euclidean data," *IEEE Signal Processing Magazine*, vol. 34, no. 4, pp. 18–42, Jul. 2017.

[7] W. L. Hamilton, R. Ying, and J. Leskovec, "Representation Learning on Graphs: Methods and Applications," *IEEE Data Eng. Bull.*, vol. 40, no. 3, pp. 52–74, 2017.

[8] H. Cai, V. Zheng, and K. Chang, "A comprehensive survey of graph embedding: Problems, techniques, and applications," *IEEE Transactions on Knowledge and Data Engineering*, vol. 30, pp. 1616–1637, 2018.

[9] M. Kinderkhedia, "Learning representations of graph data: A survey," 2019.

[10] J. Zhou, G. Cui, Z. Zhang, C. Yang, Z. Liu, and M. Sun, "Graph neural networks: A review of methods and applications," *CoRR*, 2018.

[11] Z. Wu, S. Pan, F. Chen, G. Long, C. Zhang, and P. S. Yu, "A Comprehensive Survey on Graph Neural Networks," *CoRR*, 2019.

[12] Z. Chen, F. Chen, L. Zhang, T. Ji, K. Fu, L. Zhao, F. Chen, and C. Lu, "Bridging the gap between spatial and spectral domains: A survey on graph neural networks," *CoRR*, 2020.

[13] K. Borgwardt, E. Ghisu, F. Llinares-López, L. O'Bray, and B. Rieck, "Graph kernels: State-of-the-art and future challenges," *Foundations and Trends in Machine Learning*, vol. 13, no. 5-6, pp. 531–712, 2020.

[14] G. Nikolentzos, G. Siglidis, and M. Vazirgiannis, "Graph kernels: A survey," *CoRR*, vol. abs/1904.12218, 2019.

[15] Y. Zhu, W. Xu, J. Zhang, Q. Liu, S. Wu, and L. Wang, "Deep graph structure learning for robust representations: A survey," *CoRR*, 2021.

[16] C. Zhuang and Q. Ma, "Dual graph convolutional networks for graph-based semi-supervised classification," in *Proceedings of World Wide Web Conference (WWW)*, Lyon, France, 2018, pp. 499–508.

[17] S. Cao, W. Lu, and Q. Xu, "Deep neural networks for learning graph representations," *Proceedings of the Conference on Artificial Intelligence (AAAI)*, vol. 30, no. 1, Feb. 2016.

[18] Y. Yang, H. Chen, and J. Shao, "Triplet enhanced autoencoder: Model-free discriminative network embedding," in *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence (IJCAI)*, Macao, China, August 10-16 2019, pp. 5363–5369.

[19] F. R. K. Chung, *Spectral Graph Theory*, Providence, RI, 1997.

[20] M. R. Garey and D. S. Johnson, *Computers and Intractability: A Guide to the Theory of NP-Completeness*. W. H. Freeman, 1979.

[21] D. Bieber, "The Weisfeiler-Lehman Isomorphism test," 2019. [Online]. Available: https://davidbieber.com/post/2019-05-10-weisfeiler-lehman-isomorphism-test/

[22] B. Weisfeiler and A. Leman, "The reduction of a graph to canonical form and the algebra which appears therein," *Nauchno-Technicheskaya Informatsia*, 1968.

[23] L. Babai, P. Erdös, and S. M. Selkow, "Random graph isomorphism," *SIAM J. Comput.*, vol. 9, no. 3, pp. 628–635, 1980.

[24] M. Niepert, M. Ahmed, and K. Kutzkov, "Learning Convolutional Neural Networks for Graphs," in *Proceedings of The 33rd International Conference on Machine Learning (ICML)*, vol. 48, New York, New York, USA, 20–22 Jun 2016, pp. 2014–2023.

[25] M. D. G. Mallea, P. Meltzer, and P. J. Bentley, "Capsule Neural Networks for Graph Classification using Explicit Tensorial Graph Representations," *arXiv*, 2019.

[26] H. Jin, W. Yu, and S. Li, "Graph regularized nonnegative matrix tri-factorization for overlapping community detection," *Physica A: Statistical Mechanics and its Applications*, vol. 515, no. C, pp. 376–387, 2019.

[27] M. Huang, Q. Jiang, Q. Qu, and A. Rasool, "An overlapping community detection approach in ego-splitting networks using symmetric nonnegative matrix factorization," *Symmetry*, vol. 13, no. 5, 2021.

[28] G. Ceddia, P. Pinoli, S. Ceri, and M. Masseroli, "Matrix factorization-based technique for drug repurposing predictions," *IEEE J. Biomed. Health Informatics*, vol. 24, no. 11,

pp. 3162–3172, 2020.

[29] A. Mitra, P. Vijayan, S. Parthasarathy, and B. Ravindran, "A unified non-negative matrix factorization framework for semi supervised learning on graphs," in *Proceedings of International Conference on Data Mining (SDM)*, Cincinnati, Ohio, USA, May 7-9 2020, pp. 487–495.

[30] F. Ye, C. Chen, and Z. Zheng, "Deep Autoencoder-like Nonnegative Matrix Factorization for Community Detection," in *Proceedings of the 27th International Conference on Information and Knowledge Management (CIKM)*, USA, 2018, pp. 1393–1402.

[31] J. Qiu, Y. Dong, H. Ma, J. Li, C. Wang, K. Wang, and J. Tang, "NetSMF: Large-scale network embedding as sparse matrix factorization," in *The World Wide Web Conference (WWW)*, San Francisco, CA, USA, May 13-17 2019, pp. 1509–1520.

[32] W. Jin, Y. Ma, X. Liu, X. Tang, S. Wang, and J. Tang, "Graph structure learning for robust graph neural networks," in *26th Conference on Knowledge Discovery and Data Mining (SIGKDD)*, Virtual Event, CA, USA, August 23-27 2020, pp. 66–74.

[33] L. Franceschi, M. Niepert, M. Pontil, and X. He, "Graph structure learning for GCNs," *A workshop paper at International Conference on Learning Representations (ICLR)*, 2019.

[34] W. L. Hamilton, "Graph representation learning," *Synthesis Lectures on Artificial Intelligence and Machine Learning*, vol. 14, no. 3, pp. 1–159, 2020.

[35] M. Ou, P. Cui, J. Pei, Z. Zhang, and W. Zhu, "Asymmetric transitivity preserving graph embedding," in *Proceedings of International Conference on Knowledge Discovery and Data Mining (SIGKDD)*, San Francisco, CA, USA, August 13-17 2016, pp. 1105–1114.

[36] A. Grover and J. Leskovec, "node2vec: Scalable feature learning for networks," in *Proceedings of International Conference on Knowledge Discovery and Data Mining (SIGKDD)*, 2016, pp. 855–864.

[37] B. Perozzi, R. Al-Rfou, and S. Skiena, "Deepwalk: Online learning of social representations," in *Proceedings of International Conference on Knowledge Discovery and Data Mining (SIGKDD)*, 2014, pp. 701–710.

[38] J. Tang, M. Qu, M. Wang, M. Zhang, J. Yan, and Q. Mei, "LINE: large-scale information network embedding," in *Proceedings of the 24th International Conference on World Wide Web (WWW)*, Florence, Italy, May 18-22 2015, pp. 1067–1077.

[39] J. Zhao, X. Wang, C. Shi, B. Hu, G. Song, and Y. Ye, "Heterogeneous graph structure learning for graph neural networks," in *Conference on Artificial Intelligence (AAAI)*, February 2-9 2021, pp. 4697–4705.

[40] A. Micheli, "Neural network for graphs: A contextual constructive approach," *IEEE Transactions on Neural Networks*, vol. 20, no. 3, pp. 498–511, 2009.

[41] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," in *International Conference on Learning Representations (ICLR)*, 2017.

[42] M. Zhang, Z. Cui, M. Neumann, and Y. Chen, "An end-to-end deep learning architecture for graph classification," in *AAAI*, 2018.

[43] B. Jiang, Z. Zhang, D. Lin, J. Tang, and B. Luo, "Semi-supervised learning with graph learning-convolutional networks," in *Conference on Computer Vision and Pattern Recognition (CVPR)*, Long Beach, CA, USA, June 16-20 2019, pp. 11 313–11 320.

[44] P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Liò, and Y. Bengio, "Graph attention networks," *International Conference on Learning Representations (ICLR)*, 2017.

[45] Z. Zhang, P. Cui, J. Pei, X. Wang, and W. Zhu, "Eigen-gnn: A graph structure preserving plug-in for gnns," *CoRR*, 2020.

[46] F. Errica, M. Podda, D. Bacciu, and A. Micheli, "A fair comparison of graph neural networks for graph classification," in *Proceedings of International Conference on Learning Representations (ICLR)*, 2020.

[47] D. Shuman, S. Narang, P. Frossard, A. Ortega, and P. Vandergheynst, "The emerging field of signal processing on graphs: Extending high-dimensional data analysis to networks and other irregular domains," *IEEE Signal Processing Magazine*, vol. 30, pp. 83–98, 2013.

[48] A. Sandryhaila and J. M. F. Moura, "Discrete signal processing on graphs: Graph Fourier transform," in *2013 IEEE International Conference on Acoustics, Speech and Signal Processing*, 2013, pp. 6167–6170.

[49] S. Chen, R. Varma, A. Sandryhaila, and J. Kovačević, "Discrete signal processing on graphs: Sampling theory," *IEEE Transactions on Signal Processing*, vol. 63, no. 24, pp. 6510–6523, 2015.

[50] J. Bruna, W. Zaremba, A. Szlam, and Y. LeCun, "Spectral networks and locally connected networks on graphs," in *International Conference on Learning Representations (ICLR)*, Banff, AB, Canada, April 14-16 2014.

[51] M. Defferrard, X. Bresson, and P. Vandergheynst, "Convolutional neural networks on graphs with fast localized spectral filtering," in *Advances in Neural Information Processing Systems (NIPS)*, D. Lee, M. Sugiyama, U. Luxburg, I. Guyon, and R. Garnett, Eds., vol. 29, 2016.

[52] R. Levie, F. Monti, X. Bresson, and M. M. Bronstein, "Cayleynets: Graph convolutional neural networks with complex rational spectral filters," *IEEE Transactions on Signal Processing*, vol. 67, no. 1, pp. 97–109, 2019.

[53] S. Yu, Y. Feng, D. Zhang, H. D. Bedru, B. Xu, and F. Xia, "Motif discovery in networks: A survey," *Computer Science Review*, vol. 37, p. 100267, 2020.

[54] N. Przulj, D. G. Corneil, and I. Jurisica, "Modeling interactome: scale-free or geometric?." *Bioinform.*, vol. 20, no. 18, pp. 3508–3515, 2004.

[55] N. Shervashidze, S. Vishwanathan, T. Petri, K. Mehlhorn, and K. Borgwardt, "Efficient graphlet kernels for large graph comparison," in *Proceedings of International Conference on Artificial Intelligence and Statistics (AISTAT)*, vol. 5, Clearwater Beach, Florida USA, 16–18 Apr 2009, pp. 488–495.

[56] B. Schölkopf and A. J. Smola, *Learning with kernels : support vector machines, regularization, optimization, and beyond*, 2002.

[57] P. Yanardag and S. Vishwanathan, "Deep Graph Kernels," in *Proceedings of International Conference on Knowledge Discovery and Data Mining (KDD)*, USA, 2015, pp. 1365–1374.

[58] J. Ramon and T. Gärtner, "Expressivity versus efficiency of graph kernels," in *Proceedings of European Conference on Machine Learning (ECML/PKDD)*, Cavtat-Dubrovnik, Croatia, Sep 22-23 2003, pp. 65–74.

[59] P. Mahé and J. Vert, "Graph kernels based on tree patterns for molecules," *Mach. Learn.*, vol. 75, no. 1, pp. 3–35, 2009.

[60] T. Gï¿œrtner, P. A. Flach, and S. Wrobel, "On graph kernels: Hardness results and efficient alternatives." in *COLT*, vol. 2777, 2003, pp. 129–143.

[61] R. A. Rossi, N. K. Ahmed, E. Koh, S. Kim, A. Rao, and Y. Abbasi-Yadkori, "A structural graph representation learning framework," in *International Conference on Web Search and Data Mining (WSDM)*, Houston, TX, USA, 2020, pp. 483–491.

[62] X. Li, W. Wei, X. Feng, X. Liu, and Z. Zheng, "Representation learning of graphs using graph convolutional multilayer networks based on motifs," *Neurocomputing*, vol. 464, pp. 218–226, 2021.

[63] G. Abuoda, G. D. F. Morales, and A. Aboulnaga, "Link prediction via higher-order motif features," in *Proceedings of Machine Learning and Knowledge Discovery in Databases - European Conference (ECML) (PKDD)*, vol. 11906, Würzburg, Germany, Sep 16-20 2019, pp. 412–429.
[64] B. D. McKay and A. Piperno, "Practical graph isomorphism, II," *J. Symb. Comput.*, vol. 60, pp. 94–112, 2014.
[65] P. Velickovic, "Theoretical foundations of graph neural networks," *CST Wednesday Seminar*, Feb 2021.
[66] M. Zaheer, S. Kottur, S. Ravanbakhsh, B. Poczos, R. R. Salakhutdinov, and A. J. Smola, "Deep Sets," in *Advances in Neural Information Processing Systems (NIPS)*, 2017, pp. 3391–3401.
[67] E. Wagstaff, F. Fuchs, M. Engelcke, I. Posner, and M. A. Osborne, "On the limitations of representing functions on sets," in *Proceedings of International Conference on Machine Learning (ICML)*, vol. 97, 2019, 9-15 June 2019, Long Beach, California, USA, 2019, pp. 6487–6494.
[68] H. Maron, H. Ben-Hamu, N. Shamir, and Y. Lipman, "Invariant and equivariant graph networks," in *International Conference on Learning Representations (ICLR)*, New Orleans, LA, USA, May 6-9, 2019.
[69] H. Maron, E. Fetaya, N. Segol, and Y. Lipman, "On the universality of invariant networks," in *Proceedings of International Conference on Machine Learning (ICML)*, vol. 97, Long Beach, California, USA, June, 9-15 2019, pp. 4363–4371.
[70] N. Keriven and G. Peyré, "Universal invariant and equivariant graph neural networks," in *Advances in Neural Information Processing Systems (NIPS)*, vol. 32, 2019.
[71] W. Hamilton, Z. Ying, and J. Leskovec, "Inductive Representation Learning on Large Graphs," in *Advances in Neural Information Processing Systems (NIPS)*, 2017, pp. 1024–1034.
[72] K. Xu, W. Hu, J. Leskovec, and S. Jegelka, "How Powerful are Graph Neural Networks?" in *the 7th International Conference on Learning Representations (ICLR)*, USA, May 6-9 2019.
[73] J. Lee, Y. Lee, J. Kim, A. Kosiorek, S. Choi, and Y. W. Teh, "Set transformer: A framework for attention-based permutation-invariant neural networks," in *Proceedings of International Conference on Machine Learning (ICML)*, vol. 97, Jun, 09-15 2019, pp. 3744–3753.
[74] H. Zhu, F. Feng, X. He, X. Wang, Y. Li, K. Zheng, and Y. Zhang, "Bilinear graph neural network with neighbor interactions," in *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence, (IJCAI)*, 2020, pp. 1452–1458.
[75] S. Verma and Z.-L. Zhang, "Graph Capsule Convolutional Neural Networks," *arXiv*, 2018.
[76] P. Meltzer, M. D. G. Mallea, and P. J. Bentley, "PiNet: A Permutation Invariant Graph Neural Network for Graph Classification," *CoRR*, 2019.
[77] M. Rupp, A. Tkatchenko, K.-R. Müller, and O. A. von Lilienfeld, "Fast and accurate modeling of molecular atomization energies with machine learning," *Physical Review Letters*, vol. 108, p. 058301, 2012.
[78] G. Montavon, K. Hansen, S. Fazli, M. Rupp, F. Biegler, A. Ziehe, A. Tkatchenko, A. Lilienfeld, and K.-R. Müller, "Learning invariant representations of molecules for atomization energy prediction," in *Advances in Neural Information Processing Systems (NIPS)*, vol. 25, 2012.
[79] Y. Ma, S. Wang, C. C. Aggarwal, and J. Tang, "Graph convolutional networks with eigenpooling," in *Proceedings of International Conference on Knowledge Discovery & Data Mining, (KDD)*, Anchorage, AK, USA, August, 4-8 2019, pp. 723–731.
[80] K. Zhao, Y. Rong, J. X. Yu, J. Huang, and H. Zhang, "Graph ordering: Towards the optimal by learning," *CoRR*, 2020.
[81] Z. Chen, L. Chen, S. Villar, and J. Bruna, "Can graph neural networks count substructures?" in *Advances in Neural Information Processing Systems (NeurIPS) 33*, December 6-12 2020.
[82] G. Bouritsas, F. Frasca, S. Zafeiriou, and M. M. Bronstein, "Improving graph neural network expressivity via subgraph isomorphism counting," *CoRR*, 2020.
[83] D. Floros, N. Pitsianis, and X. Sun, "Fast graphlet transform of sparse graphs," in *2020 IEEE High Performance Extreme Computing Conference, (HPEC)*, Waltham, MA, USA, Sep 22-24 2020, pp. 1–8.
[84] J. Gilmer, S. S. Schoenholz, P. F. Riley, O. Vinyals, and G. E. Dahl, "Neural message passing for quantum chemistry," in *Proceedings of International Conference on Machine Learning, ICML*, vol. 70, Sydney, NSW, Australia, August, 6-11 2017, pp. 1263–1272.
[85] V. Arvind, F. Fuhlbrück, J. Köbler, and O. Verbitsky, "On Weisfeiler-Leman invariance: Subgraph counts and related graph properties," in *Proceedings of Fundamentals of Computation Theory (FCT)*, vol. 11651, Copenhagen, Denmark, August 12-14 2019, pp. 111–125.
[86] R. L. Murphy, B. Srinivasan, V. A. Rao, and B. Ribeiro, "Janossy pooling: Learning deep permutation-invariant functions for variable-size inputs," in *International Conference on Learning Representations (ICLR)*, New Orleans, LA, USA, May, 6-9 2019.
[87] B. Bloem-Reddy and Y. W. Teh, "Probabilistic symmetries and invariant neural networks," *J. Mach. Learn. Res.*, vol. 21, pp. 90:1–90:61, 2020.
[88] R. Murphy, B. Srinivasan, V. Rao, and B. Ribeiro, "Relational pooling for graph representations," in *Proceedings of International Conference on Machine Learning (ICML)*, vol. 97, 09–15 Jun 2019, pp. 4663–4673.
[89] E. Cohen-Karlik, A. B. David, and A. Globerson, "Regularizing towards permutation invariance in recurrent models," in *Advances in Neural Information Processing Systems (NIPS)*, Dec, 6-12 2020.
[90] J. Moore and J. Neville, "Deep collective inference," in *Proceedings of Conference on Artificial Intelligence (AAAI)*, San Francisco, California, USA, Feb, 4-9 2017, pp. 2364–2372.

Tuyen Ho-Thi-Thanh is a PhD student in Computer Science at the University of Science – Vietnam National University, Ho Chi Minh City, Vietnam. She has been teaching courses related to Computer Science program since 2013. She is currently a lecturer and a member of strong research team at the School of Technology and Design - University of Economics Ho Chi Minh City. Her main research includes artificial intelligence, graph structure learning and graph neural network.
Email: tuyenhtt@ueh.edu.vn

Improving the Heuristic Algorithms to Solve a Steiner-minimal-tree Problem in Large Size Sparse Graphs

Tran Viet Chuong[1], Phan Tan Quoc[2], Ha Hai Nam[3]

[1] Information Technology & Communication Center, Information & Communication Department of Ca Mau Province, Vietnam
[2] Faculty of Information Technology, Saigon University, Vietnam
[3] Vietnam National Institute of Software and Digital Content Industry (NISCI), Ministry of Information and Communications

Correspondence: Tran Viet Chuong, chuongcm74@gmail.com
Communication: received 5 January 2022, revised 23 January 2022, accepted 28 February 2022
DOI: 10.32913/mic-ict-research-vn.v2022.n1.1029

***Abstract*: Steiner Minimal Tree (SMT) is a combinatorial optimization problem that has many important applications in science and engineering; this is an NP-hard class problem. In recent decades, there have been a series of scientific papers published for solving the SMT problem using the approaches from exact solutions (such as dynamic programming, branch and bound) and approximate solutions (such as heuristic algorithm, metaheuristic algorithm). This paper proposes an improvement for two heuristic algorithms, PD-Steiner and SPT-Steiner, to solve a SMT problem in large size sparse graphs with edge weights not exceeding 10, and validates this proposal on large-size sparse graphs up to 100000 vertices. These experimental results are useful information for further research on the SMT problem.**

Keywords: *Steiner minimal tree, sparse graph, heuristic algorithm, metaheuristic algorithm.*

I. INTRODUCTION

A. Definitions

This section contains definitions and properties pertaining to the SMT problem.

Definition 1: Steiner tree [3]

Given $G = (V(G), E(G))$ is an interconnected undirected single graph with non-negative edge weights; where $V(G)$ is a set of n vertices, $E(G)$ is a set of m edges, $w(e)$ is the weight of edge e, $e \in E(G)$. A Steiner tree of L is a tree T that connects all vertices in a subset L of $V(G)$.

L is called a terminal set, its vertices are called terminal vertices, and vertices belonging to T but not L, are called Steiner vertices. Unlike other spanning tree problems, the Steiner tree only needs to traverse all of the vertices in the terminal set L and possibly some more vertices in the set $V(G)$.

Definition 2: The cost of Steiner tree [3]

Given that $T = (V(T), E(T))$ is a Steiner tree of graph G, the cost of tree T, denoted $C(T)$, is the total weight of the tree's edges, or $C(T) = \sum_{e \in E(T))} w(e)$.

Definition 3: Steiner Minimal Tree [3]

Given the graph G described above, the problem of establishing a Steiner tree at the lowest possible cost is called the Steiner Minimal Trees problem – *SMT*; or, more succinctly, Steiner Trees problem.

SMT is a combinatorial optimization problem in graph theory. In general, SMT has been proven to belong to the NP-hard problem [3, 13]. There are two special cases for *SMT* problem that can be solved in polynomial time: When $L = V(G)$ and $|L| = 2$ ($|L|$ denotes the number of vertices in the set L): When $L = V$, *SMT* problem can be solved by the smallest spanning tree algorithms; such as Prim [30], Kruskal [30]; when $|L| = 2$, *SMT* problem can be solved by algorithms of finding shortest paths between two vertices, such as Dijkstra [30].

For brevity, in this article, the *graph* is understood as a single, undirected, interconnected graph, with non-negative weights.

Example 1:

Given a graph G with 9 vertices and 10 edges as shown in Figure 1 and *terminal* set $L = \{2, 8, 9\}$.

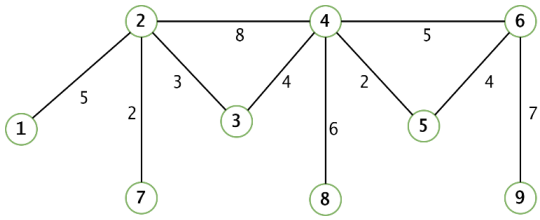


Figure 1. Illustration of a weighted interconnected undirected graph G.

The Steiner Minimal Tree founding corresponding to the *terminal* set L on graph G is T with $V(T) = \{2, 3, 4, 6, 8, 9\}$ and $E(T) = \{(2, 3), (3, 4), (4, 6), (4, 8), (6, 9)\}$ as illustrated in Figure 2; tree T has the Steiner vertices set $\{3, 4, 6\}$ and the cost of 25.

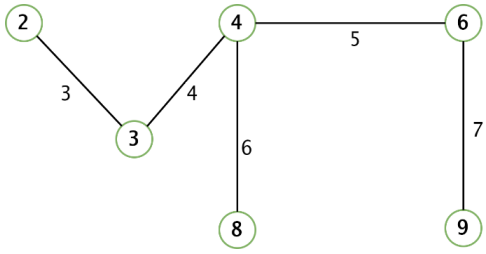


Figure 2. Steiner Minimal Tree corresponding to terminal set L of graph G.

B. Application of SMT problem

SMT problems can be found in important applications in several scientific and engineering fields such as the communication network design problems [2, 6, 11, 33, 41], VLSI design problems (Very Large Scale Integrated) [10], and problems related to network systems with the lowest cost [8, 10, 12, 18, 20, 23], etc.

C. Some forms of Steiner Minimal Tree problems

Steiner Minimal Tree problems are currently being studied in the following forms:

The first one is the Steiner Tree problem with Euclidean distance [13, 17, 38]. In the OXY coordinate plane, given a graph $G = (V(G), E(G))$ and a vertex set $Y \subseteq V$. We need to find trees passing through all the vertices in set Y and have a minimum total length, allowing us to add several sub-points (Steiner points) taken from the vertices of V. Euclide distance between two points $P_1(x_1, y_1)$ and $P_2(x_2, y_2)$ is defined as the length of the line connecting the two vertices P_1, P_2.

The second form is the Steiner Tree problem with rectilinear distance [45]. In the OXY coordinate plane, given a graph $G = (V(G), E(G))$ and a vertex set $Y \subseteq V$ We need to find trees passing through all the vertices in the set Y and have a minimum total length, allowing us to add a number of sub-points (Steiner points) taken from the vertices of V. $d(P_1, P_2) = |x_1 - x_2| + |y_1 - y_2|$ is the rectilinear distance between two points $P_1(x_1, y_1)$ and $P_2(x_2, y_2)$.

The third form is the Steiner Tree problem, with the distance between vertices defined as the weights of edges (random distance values) [4, 24], which has been chosen as the scope of this paper's research on the Steiner Minimal Tree problem.

D. Some SMT studies and problems that need to be solved

Due to its scientific nature and wide applicability, SMT problems have attracted the attention of many scientists in the world for decades [27, 35, 44]. Some of them are Martin Zachariasen [21], Tobias Polzin [37], Pieter Oloff De Wet [26], Xinhui Wang [40], Jon William Van Laarhoven [17], and Zhiliu Zhang [45].

Currently, the publications on SMT problems can be divided into the following approaches: graph reduction algorithms, algorithms to find the exact solution, algorithms to find approximate solutions, heuristic algorithms, and metaheuristic algorithms.

***The first approach*: graph reduction algorithms**

There are some works on graph reduction for Steiner Tree problems regarding the techniques to minimize the graph size such as the works of Jeffrey H. Kingston and Nicholas Paul Sheppard [16], Thorsten Koch and Alexander Martin [36], C. C. Ribeiro and M.C. Souza [5], etc.

The general concept of graph reduction algorithms is aimed at two goals. The first one is to increase the number of vertices in the terminal set; the second one is to remove vertices of the graph that are certainly not on the Steiner Minimal Tree that has been sought.

The quality of algorithms solving the SMT problems depends on the size of the coefficient $n - |L|$. Therefore, the purpose of graph reduction algorithms is to minimize the coefficient $n - |L|$.

The graph reduction algorithms are considered an important data preprocessing step to improve the quality of solving SMT problems. This step has become more and more necessary for algorithms to find the exact solutions, such as dynamic programming or branch and bound.

***The second approach*: algorithms to find the exact solution**

Some studies to find the exact solution to an SMT problem are known as the dynamic programming algorithm of Dreyfus and Wagner [32], an algorithm based on the Lagrange relaxation of Beasley [15], branch and bound algorithms of Koch and Martin [36], and Xinhui Wang [40, 25].

Although this approach can help find the exact solution, it can only solve small-scale problems. As a result, their utility is limited. Solving the exact SMT problem is really a challenge in combinatorial optimization theory [22, 38].

This approach is an important basis for evaluating the solution quality of other approximate algorithms when solving SMT problems.

***The third approach*: α - approximation algorithms.**

The α - approximation algorithm aims to find an approximated solution with α ratio to the optimal solution [3].

The advantage of this approximation algorithm is that it is mathematically guaranteed in the sense mentioned above. Meanwhile, its drawback is that the approximate ratio found in practice is often much worse than the quality of solutions found by other approximate algorithms based on experimentation. The MST-Steiner algorithm of Bang Ye Wu and Kun-Mao Chao has the ratio of 2 [3], while the Zelikovsky-Steiner algorithm has the ratio of 11/6 [3]; The best ratio found for the SMT problem is currently 1.39 [7, 29, 30, 34].

***The fourth approach*: heuristic algorithms**

The heuristic algorithm is the specific experience of finding a solution to a particular optimization problem. The heuristic algorithm often finds an acceptable solution in the time allowed, but is uncertain whether it is the exact solution. Heuristic algorithms are also unlikely to be effective on all types of data for a particular problem.

The typical heuristic algorithms for SMT problems are SPH [5], Heu [36], distance network heuristic of Kou, and Markowsky and Berman [19].

***The fifth approach*: metaheuristic algorithms**

The metaheuristic algorithm uses several heuristics combined with auxiliary techniques to explore the search space. It belongs to the class of optimal search algorithms [42, 43].

There have been a lot of works using metaheuristic algorithms to solve SMT problems such as the Variable neighbor search algorithm, the Hill climbing search algorithm, the Genetic algorithms [9], the Tabu search algorithm [5, 39], the Parallel genetic algorithm (PGA) [1, 24], and the Bees algorithm.

Having considered various types of approaches, the authors of this paper would like to propose an improvement for the heuristic algorithms to solve the SMT problem in the case of large sparse graphs, giving an acceptable quality of the solution and a faster running time than other known heuristic algorithms. When applied to real-world problems, this proposed method is more meaningful.

E. Experimental data system

Definition 4: Sparse graphs

According to the conventional definition (at the URL: https://www.baeldung.com/cs/graphs-sparse-vs-dense), a simple graph with n vertices is a sparse graph with no more than $n(n-1)/4$ edges.

To experiment with related algorithms, most of the works use 78 datasets which are sparse graphs in the standard experimental data system for the Steiner tree problem (at the URL: http://people.brunel.ac.uk/~mastjjb/jeb/orlib/steininfo.html [14]) which contain the following information about graph groups:

Steinb graphs have 50 to 100 vertices, and 63 to 200 edges; steinb contains a total of 18 graphs. Steinc graphs have 500 vertices, and 625 to 12500 edges; steinc contains a total of 20 graphs. Steind graphs have 1000 vertices, and 1250 to 25000 edges; steind contains a total of 20 graphs. The steine graphs have 2500 vertices, and 3125 to 62500 edges; steine graph contains a total of 20 graphs.

TABLE I
INFORMATION ON GROUPS OF GRAPHS

Groups of graphs	n	m	Number of tests
steinb.txt	50..100	63..200	18
steinc.txt	500	625..12500	20
steind.txt	1000	1250..25000	20
steine.txt	2500	3125..62500	20

Because the problem belongs to the *NP-hard* class, the application of the *SMT* problem should be considered from either the perspective of design when priority is given to the quality of the solution, or the perspective of implementation when run-time is referred. The purpose of this paper is to improve the heuristic algorithms to solve the *SMT* problem in the case of large sparse graphs with acceptable solution quality and shorter running time when compared to currently available heuristic algorithms. This proposed method is more meaningful when considering the perspective of the implementation when applying the Steiner tree problem to reality. Experiments with the improved algorithms were conducted on a data system with 80 graphs ranging in size from n to 100000 vertices.

II. HEURISTIC ALGORITHMS FOR SOLVING AN SMT PROBLEM

This paper proposes an improvement for 2 heuristic algorithms PD-Steiner and SPT-Steiner [4] based on the idea combination of finding the shortest path and the smallest spanning tree of the graph to solve SMT problems in large size sparse graphs with an edge weight of less

than 10. Because two heuristic algorithms SPT-Steiner, PD-Steiner [4], and other currently known heuristics do not solve this problem, the proposed ones are called i-PD-Steiner and i-SPT-Steiner. The authors of this paper conducted experiments and compared the quality of the i-PD-Steiner algorithm and the i-SPT-Steiner algorithm with MST-Steiner algorithm [3]. These algorithms have input data as graph $G = (V, E, w)$, *terminal* vertices set $L \subseteq V$; and the output is the cost of the Steiner tree T.

A. MST-Steiner algorithm

Algorithm 1: MST-Steiner algorithm of Bang Ye Wu and Kun-Mao Chao has the ratio of 2 [3]

Inputs: Graph $G = (V, E, w)$ and a terminal set $L \subset V$
Outputs: Steiner tree T
begin
  Construct the metric closure G_L on the terminal set L. G_L is the complete graph in which each edge is weighted by the shortest path between the nodes in G.;
  Find a minimum spanning tree T_L on G_L.;
  T $\leftarrow \emptyset$.;
  for *each edge* $e = (u, v) \in E(T_L)$ **do**
    Find a shortest path P from u to v on G.;
    if P *contains less than two vertices in* T **then**
      Add P to T;
    else
      Let p_i and p_j be the first and the last vertices already in T;
      Add sub-paths from u to p_i and from p_j to v to T;
    end
  end
  return T;
end

MST-Steiner algorithm has the complexity $O(n|L|^2)$ [3].

B. i-SPT-Steiner algorithm

Definition 5: The shortest path tree [3]

Given a graph G, Shortest Path Tree (SPT) with the origin at vertex s is the spanning tree of G with a set of edges that is on the shortest paths starting from vertex s to the remaining vertices of G.

The shortest path tree originating at the vertex s of graph G can be found by applying Dial algorithm [28] (a variant of Dijkstra algorithm is recommended for small edge weights) to find the shortest path from vertex s to all remaining vertices.

In the experimental data system of this paper, the maximum edge weight is 10. The complexity of Dial algorithm is $O(m + n * C)$ [28] where C is the maximum weighted value of the graph.

Algorithm 2: i-SPT-Steiner (improved-Shortest Path Tree-Steiner)

Inputs: Graph $G = (V, E, w)$ and a terminal set $L \subset V$
Outputs: Steiner tree T
begin
  Find the shortest path trees starting at the vertices of the set V: SPT_1, SPT_2, ..., $SPT_{|v|}$, each SPT_i only need to find the shortest path tree containing all vertices $v \in L$ (In this step, use Dial algorithm [28] to find the shortest paths instead of using the Dijkstra algorithm as in SPT-Steiner heuristic [4]).;
  For each tree $T = SPT_i$, traversal pendants $u \in V(T)$, if $u \notin L$ and u is a pendant vertex, delete the edge containing u from $E(T)$, delete vertex u in $V(T)$ and update the degree of the vertex adjacent to the vertex u in T. Repeat this step until T no longer changes (this step is called deleting redundant edges - the edge contains a pendant vertex u); after this step, the $SPT_{i'}$ is obtained, corresponding to SPT_i.;
  In the process of deleting redundant edges, the queue is used to contain vertices. These vertices are not in set L, and the priority is the vertex with small degree. This technique of using the queues during this phase minimizes the time to remove redundant edges. ;
  Find a $SPT_{i'}$ with the lowest total weight.;
  return T;
end

The complexity for i-SPT-Steiner algorithm

Since Dial algorithm uses a queue data structure with time complexity of $O(m + n * C)$ [28], step 1 has the complexity of $O(n*(m+n*C))$, step 2 has the complexity of $O(n^2)$, and step 3 has the complexity of $O(n)$. As a result, the i-SPT-Steiner algorithm has time complexity of $O(n * (m + n * C))$.

Comparing with the SPT-Steiner algorithm [4], each shortest path tree must traverse through all the vertices of the set V, so the SPT-Steiner algorithm has a much longer running time. However, these have been improved in the i-SPT-Steiner algorithm.

In the i-SPT-Steiner algorithm, it is only required to find the shortest path tree that contains all the vertices in the set L. This is due to the fact that the remaining vertices that have not been traversed are definitely redundant and will be removed when the deleting redundant edges step (in step 2) is performed. This will help to reduce the running time of the program (about 2 - 3 times faster with our experimental data). Furthermore, a priority queue data structure is used to reduce the complexity of the step of deleting redundant edges.

Example 2: Illustration of step-by-step implementation of the i-SPT-Steiner algorithm

Given a graph G with 10 vertices and 15 edges as Figure 3; where set $L = \{1, 5, 6\}$.

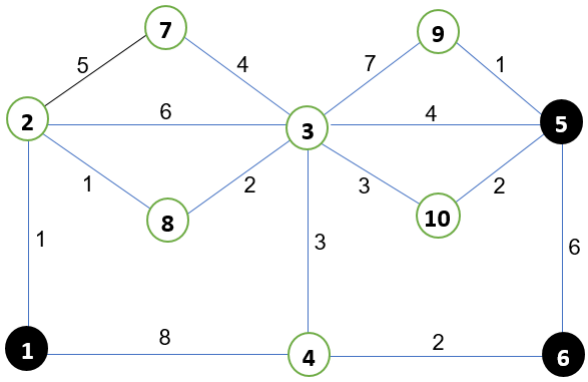


Figure 3. Weighted interconnected undirected graph G

According to the definition 4 and the i-SPT-Steiner algorithm, we construct the shortest-path trees originating at vertices **1, 2, 3, 4, 5, 6, 7, 8, 9,** and **10**. After deleting the edges, we have Steiner trees with values of 13, 13, 13, 13, 14, 15, 19, 13, 14, and 15 respectively.

We consider Figure 3. Firstly, the shortest path tree rooted at vertex **1** need to be found; as shown in Figure 4. Next, the excess edges **(2, 7), (5, 9), (3, 10)** are removed, and a Steiner tree with a cost of 13 is obtained, as shown in Figure 5.

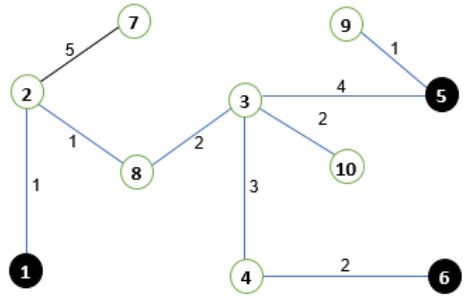


Figure 4. The shortest path tree rooted at vertex **1**

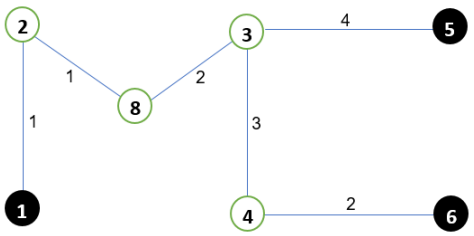


Figure 5. Steiner tree after redundant edges have been deleted

According to the above i-SPT-Steiner algorithm, the found Steiner tree costs 13, as shown in Figure 5.

Algorithm 3: i-PD-Steiner (improved-Prim + Dijkstra-Steiner)

Inputs: Graph $G = (V, E, w)$ and a terminal set $L \subset V$
Outputs: Steiner tree T
begin
  Select a vertex $u \in L$; set $V(T) = \{u\}$;
  for *each vertex* $v \in L$ **do**
    Find the shortest path tree from vertex v;
    (In this step, use Dial algorithm [28] to find the shortest paths instead of using the Dijkstra algorithm as in PD-Steiner heuristic [4]);
  end
  while T *does not contain all vertices in the set* L **do**
    From each vertex $v \in L$ and v is not already in T, select the shortest path P from vertex v to vertex $z \in T$, with z as the vertex of the shortest path of all paths starting from vertex v to vertices of $V(T)$;
    Add vertices and edges on path P to tree T and ignore the vertices and edges already in T;
  end
  return T;
end

C. i-PD-Steiner algorithm

The complexity for i-PD-Steiner algorithm

Line 1 has the complexity of $O(1)$;

Line 3 Due to using the Dial algorithm to find the shortest path tree, the complexity is $O(m+n*C)$. As a result, the loop in **line 2** has complexity of $O(|L| * (m + n * C))$.

Line 6 has complexity of $O(|L|*|T|)$, so the loop in **line 5** has complexity of $O(|L|^2 * |T|)$.

Altogether, the i-PD-Steiner algorithm has time complexity: $O(|L| * max(m + n * C, |L| * |T|))$.

Example 3: Illustration of step-by-step implementation of the i-PD-Steiner algorithm

The consideration of Figure 3 makes an illustration of the step-by-step implementation if i-PD-Steine algorithm is applied. Firstly, choose $V(T) = \{\mathbf{1}\}$, the shortest path from vertex **5** to vertex **1** is shown in Figure 6 (vertex **5** is chosen because it's a vertex in L, not in $V(T)$ which has the shortest path to a vertex in T). The next step is to find the shortest paths from vertex **6** to vertices **1, 2, 3, 5,** and **8**. The ones found have values of **9, 8, 5, 9, and 7** respectively. After that, the shortest path from vertex **6** to vertex **3** is selected. The found Steiner tree is shown in Figure 7 at a cost of 13.

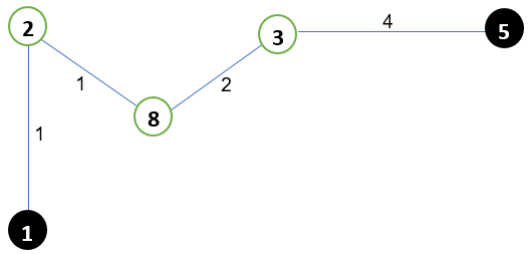


Figure 6. The shortest path from vertex **5** to vertex **1**

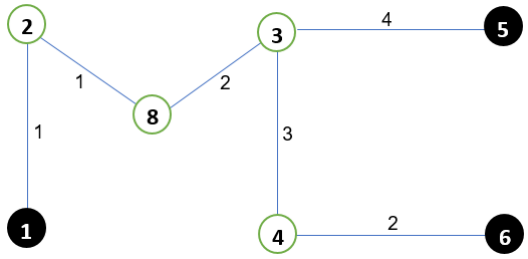


Figure 7. The shortest path from vertex **6** to vertex **3**

And according to the above i-PD-Steiner algorithm, the found Steiner tree costs 13, such as in Figure 7.

III. EXPERIMENT AND EVALUATION

A. Experimental data

Since using the heuristic algorithms to approach SMT problems has more advantages than using the metaheuristic algorithms for the large size graphs, at least in terms of timing, 80 datasets are recommended. Of them, 20 test sets are sparse graphs with 10000 vertices that are randomly generated and named *steinf1.txt, steinf2.txt, ..., steinf20.txt*; 20 test sets are sparse graphs with 20000 vertices that are randomly generated and named *steing1.txt, steing2.txt,..., steing20.txt*; 20 test sets are sparse graphs with 50000 vertices randomly generated and named *steinh1.txt, steinh2.txt,..., steinh20.txt* and 20 test sets are sparse graphs with 100000 vertices that are randomly generated and named *steini1.txt, steini2.txt,..., steini20.txt*.

These graphs are generated first from a random spanning tree that traverses all n vertices of the graphs; then, the edges are randomly generated until the graph has m edges; Edge weights are also randomly generated and have a maximum value of 10.

B. Experimental environment

MST-Steiner, i-SPT-Steiner, i-PD-Steiner algorithms are installed in C++17 using the Code::Blocks 17.12 environment and GNU GCC ver 9.3.0 compiler. They are also tested on a virtual server running Ubuntu 20.04.1 LTS (Focal Fossa), 64-bit, Intel (R) Xeon (R) Platinum 8160 @ 2.8 GHz CPU, 16 cores, 33MB Cache, and 64GB RAM.

Compile command that has been used for the experiment:

g++ -std=c++17 –O2 -o MST_STEINER MST_STEINER.cpp

g++ -std=c++17 –O2 -o SPT_STEINER SPT_STEINER.cpp

g++ -std=c++17 –O2 -o PD_STEINER PD_STEINER.cpp

In particular, most of the data structures have been changed so as to become more dynamic and optimal data structures. As a result, the experiment can be tested with large sparse graphs with up to 100000 vertices.

C. Experimental results and evaluation

The experimental results of the algorithms are recorded in Table II, Table III, Table IV, and Table V. These tables follow the hereafter patterns and labels. The first column (Test) is the name of the datasets in the experimental data system; the number of vertices (n), edges (m) and vertices in the *terminal* set ($|L|$) of each graph. The subsequent columns record the value of the Steiner tree cost for each algorithm.

TABLE II
THE EXPERIMENTAL RESULT OF THE ALGORITHMS ON STEINF GRAPH GROUP

Test	n	m	L	MST-Steiner	i-SPT-Steiner	i-PD-Steiner
steinf1.txt	10000	93750	10	58	49	51
steinf2.txt	10000	93750	20	97	95	94
steinf3.txt	10000	93750	834	2119	2551	2027
steinf4.txt	10000	93750	1250	2801	3532	2691
steinf5.txt	10000	93750	2500	4957	6567	4826
steinf6.txt	10000	125000	10	40	40	39
steinf7.txt	10000	125000	20	71	67	66
steinf8.txt	10000	125000	834	1961	2399	1838
steinf9.txt	10000	125000	1250	2576	3303	2444
steinf10.txt	10000	125000	2500	4282	5823	4124
steinf11.txt	10000	156250	10	35	35	35
steinf12.txt	10000	156250	20	77	74	70
steinf13.txt	10000	156250	834	1727	2199	1627
steinf14.txt	10000	156250	1250	2335	3049	2243
steinf15.txt	10000	156250	2500	3933	5398	3797
steinf16.txt	10000	187500	10	43	38	38
steinf17.txt	10000	187500	20	75	69	66
steinf18.txt	10000	187500	834	1597	2050	1513
steinf19.txt	10000	187500	1250	2244	2910	2153
steinf20.txt	10000	187500	2500	3764	5122	3624

Evaluating 20 datasets on *steinf* group displayed in Table II leads to certain comparative values in terms of quality. i-PD-Steiner algorithm gives *better* solution quality, *equivalent, less than* MST-Steiner algorithm with the percentages of 95.0%, 5.0%, and 0.0%; i-SPT-Steiner algorithm gives *better* solution quality, *equivalent, less than* MST-Steiner algorithm with the percentages of 30.0%, 10.0%, and

60.0%; and i-PD-Steiner algorithm gives *better* solution quality, *equivalent, less than* i-SPT-Steiner algorithm with the percentages of 85.0%, 10.0% and 5.0%.

The comparison of algorithms on steinf datasets are illustrated by the chart as shown in Figure 8.

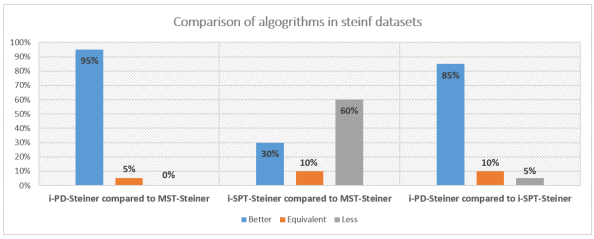


Figure 8. The comparison of algorithms on *steinf* datasets

TABLE III
THE EXPERIMENTAL RESULT OF THE ALGORITHMS ON STEING GRAPH GROUP

Test	n	m	L	MST-Steiner	i-SPT-Steiner	i-PD-Steiner
steing1.txt	20000	215000	15	79	75	75
steing2.txt	20000	215000	25	110	102	105
steing3.txt	20000	215000	950	2426	3081	2348
steing4.txt	20000	215000	1750	4039	5085	3838
steing5.txt	20000	215000	3780	7318	9689	7099
steing6.txt	20000	275000	15	71	66	64
steing7.txt	20000	275000	25	112	111	100
steing8.txt	20000	275000	950	2245	2830	2129
steing9.txt	20000	275000	1750	3643	4709	3489
steing10.txt	20000	275000	3780	6511	8773	6319
steing11.txt	20000	385000	15	65	60	59
steing12.txt	20000	385000	25	98	94	90
steing13.txt	20000	385000	950	2041	2577	1919
steing14.txt	20000	385000	1750	3211	4253	3081
steing15.txt	20000	385000	3780	5788	7981	5641
steing16.txt	20000	447500	15	61	55	54
steing17.txt	20000	447500	25	95	93	87
steing18.txt	20000	447500	950	1954	2485	1845
steing19.txt	20000	447500	1750	3079	4063	2970
steing20.txt	20000	447500	3780	5534	7647	5380

Evaluating 20 datasets on *steing* group displayed in Table III leads to certain comparative values in terms of quality. i-PD-Steiner algorithm gives *better* solution quality, *equivalent, less than* MST-Steiner algorithm with the percentages of 100.0%, 0.0%, and 0.0%; i-SPT-Steiner algorithm gives *better* solution quality, *equivalent, less than* MST-Steiner algorithm with the percentages of 40.0%, 0.0%, and 60.0%; and i-PD-Steiner algorithm gives *better* solution quality, *equivalent, less than* i-SPT-Steiner algorithm with the percentages of 90.0%, 5.0% and 5.0%.

The comparison of algorithms on steing datasets are illustrated by the chart as shown in Figure 9.

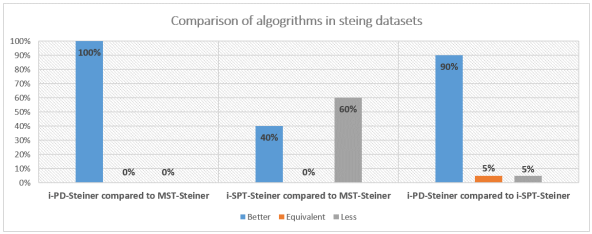


Figure 9. The comparison of algorithms on *steing* datasets

TABLE IV
THE EXPERIMENTAL RESULT OF THE ALGORITHMS ON STEINH GRAPH GROUP

Test	n	m	L	MST-Steiner	i-SPT-Steiner	i-PD-Steiner
steinh1.txt	50000	425000	15	76	71	71
steinh2.txt	50000	425000	25	129	116	118
steinh3.txt	50000	425000	950	2707	3296	2587
steinh4.txt	50000	425000	1750	4344	5466	4125
steinh5.txt	50000	425000	3780	7947	10348	7563
steinh6.txt	50000	475000	15	71	70	66
steinh7.txt	50000	475000	25	99	105	96
steinh8.txt	50000	475000	950	2531	3129	2402
steinh9.txt	50000	475000	1750	4198	5299	3951
steinh10.txt	50000	475000	3780	7718	10099	7336
steinh11.txt	50000	528000	15	80	72	74
steinh12.txt	50000	528000	25	112	110	106
steinh13.txt	50000	528000	950	2498	3082	2378
steinh14.txt	50000	528000	1750	4026	5132	3784
steinh15.txt	50000	528000	3780	7275	9702	6939
steinh16.txt	50000	587500	15	70	67	63
steinh17.txt	50000	587500	25	123	109	109
steinh18.txt	50000	587500	950	2408	2962	2262
steinh19.txt	50000	587500	1750	3886	4987	3643
steinh20.txt	50000	587500	3780	7127	9489	6797

Evaluating 20 datasets on *steinh* group displayed in Table IV leads to certain comparative values in terms of quality. i-PD-Steiner algorithm gives *better* solution quality, *equivalent, less than* MST-Steiner algorithm with the percentages of 100.0%, 0.0%, and 0.0%; i-SPT-Steiner algorithm gives *better* solution quality, *equivalent, less than* MST-Steiner algorithm with the percentages of 35.0%, 0.0%, and 65.0%; and i-PD-Steiner algorithm gives *better* solution quality, *equivalent, less than* i-SPT-Steiner algorithm with the percentages of 80.0%, 10.0% and 10.0%.

The comparison of the algorithms on steinh datasets are illustrated by the chart as shown in Figure 10.

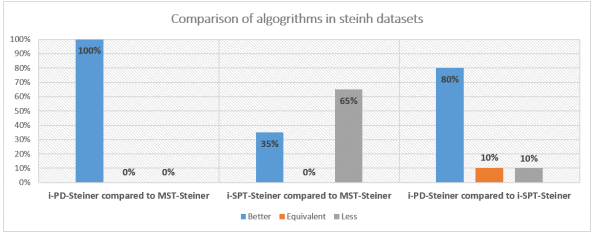


Figure 10. The comparison of algorithms on *steinh* datasets

TABLE V
THE EXPERIMENTAL RESULT OF THE ALGORITHMS ON STEINI GRAPH GROUP

Test	n	m	L	MST-Steiner	i-SPT-Steiner	i-PD-Steiner
steini1.txt	100000	125000	25	188	190	180
steini2.txt	100000	125000	45	369	364	358
steini3.txt	100000	125000	1250	6390	6677	6363
steini4.txt	100000	125000	2450	12209	12810	12178
steini5.txt	100000	125000	4500	21754	22868	21664
steini6.txt	100000	200000	25	148	144	142
steini7.txt	100000	200000	45	264	273	253
steini8.txt	100000	200000	1250	4978	5510	4902
steini9.txt	100000	200000	2450	9066	10376	8969
steini10.txt	100000	200000	4500	15126	17456	14879
steini11.txt	100000	500000	25	111	110	104
steini12.txt	100000	500000	45	184	191	173
steini13.txt	100000	500000	1250	3133	3849	2974
steini14.txt	100000	500000	2450	5487	6889	5207
steini15.txt	100000	500000	4500	8951	11629	8532
steini16.txt	100000	2500000	25	66	72	66
steini17.txt	100000	2500000	45	120	128	111
steini18.txt	100000	2500000	1250	2031	2556	1951
steini19.txt	100000	2500000	2450	3366	4529	3268
steini20.txt	100000	2500000	4500	5167	7591	5118

Evaluating 20 datasets on *steini* group displayed in Table V leads to certain comparative values in terms of quality. i-PD-Steiner algorithm gives *better* solution quality, *equivalent, less than* MST-Steiner algorithm with the percentages of 95.0%, 5.0%, and 0.0%; i-SPT-Steiner algorithm gives *better* solution quality, *equivalent, less than* MST-Steiner algorithm with the percentages of 15.0%, 0.0%, and 85.0%; and i-PD-Steiner algorithm gives *better* solution quality, *equivalent, less than* i-SPT-Steiner algorithm with the percentages of 100.0%, 0.0% and 0.0%.

The comparison of the algorithms on steini datasets are illustrated by the chart as shown in Figure 11.

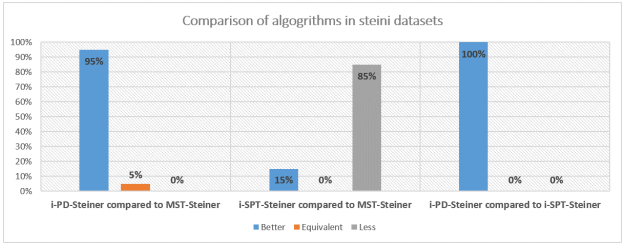


Figure 11. The comparison of algorithms on *steini* datasets

D. Evaluation of experimental results

Evaluating 80 datasets leads to certain comparative values in terms of quality. i-PD-Steiner algorithm gives *better* solution quality, *equivalent*, *less than* MST-Steiner algorithm with the percentages of 97.5%, 2.5%, and 0.0%; i-SPT-Steiner algorithm gives *better* solution quality, *equivalent*, *less than* MST-Steiner algorithm with the percentages of 30.0%, 2.5%, and 67.5%; and i-PD-Steiner algorithm gives *better* solution quality, *equivalent*, *less than* i-SPT-Steiner algorithm with the percentages of 88.75%, 6.25% and 5.0%. We noted the average run time of three algorithms i-PD-Steiner, MST-Steiner and i-SPT-Steiner on datasets as shown in Table VI.

TABLE VI
AVERAGE RUNNING TIME OF ALGORITHMS (UNIT IN SECONDS)

Groups of graphs	i-PD-Steiner	MST-Steiner	i-SPT-Steiner
steinf.txt	83,758	137,538	928,68
steing.txt	280,356	421,402	2374,775
steinh.txt	416,057	766,585	30416,253
steini.txt	1243,415	1561,921	93943,235

In other words, i-PD-Steiner algorithm has a much faster runtime than the MST-Steiner and i-SPT-Steiner algorithms. The runtime depends on not only the time complexity of the algorithm but also the experimental environment. Besides, the information about the runtime of the algorithms displayed above also provides reliable reference information about these algorithms (The queue data structure has been chosen to install the above algorithms).

The comparison of i-PD-Steiner, i-SPT-Steiner, and MST-Steiner algorithms on a total of 80 datasets are illustrated by the chart as shown in Figure 12.

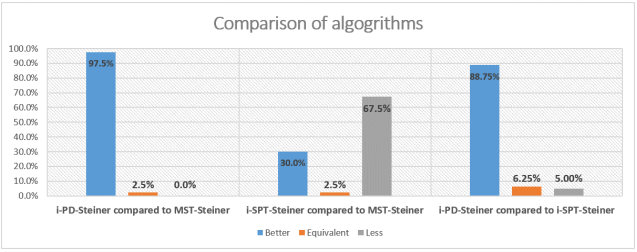


Figure 12. The comparison of the algorithms on 80 datasets

IV. CONCLUSION

The authors of this paper proposed the improvement to the heuristic algorithms SPT-Steiner and PD-Steiner [4] to solve the SMT problem in large size sparse graphs with edge weight not exceeding 10. Experiments were set up and evaluation was made in details on 80 datasets of sparse graphs with large sizes up to 100000 vertices. i-PD-Steiner algorithm gives *better* quality solution or *equivalent* MST-Steiner algorithm on 100.0% datasets. i-SPT-Steiner algorithm gives *better* quality solution or *equivalent* MST-Steiner algorithm on 32.5% datasets. i-PD-Steiner algorithm gives *better* quality solution or *equivalent* i-SPT-Steiner algorithm above 95.0% of the data. The runtime of

the i-PD-Steiner algorithm is much shorter than the one of the i-SPT-Steiner algorithm. The two heuristic algorithms i-SPT-Steiner and i-PD-Steiner as well as the experimental results in this paper are useful information for further studies on the SMT problem, especially for solving the SMT problems in the case of large sparse graphs.

REFERENCES

[1] Ankit Anand, Shruti, Kunwar Ambarish Singh, "An efficient approach for Steiner tree problem by genetic algorithm", *International Journal of Computer Science and Engineering (SSRG-IJCSE)*, vol.2, 2015, pp.233-237.

[2] Arie M.C.A. Koster, Xavier Munoz (Eds), "Graphs and algorithms in communication networks", *Springer*, 2010, pp.1-177.

[3] Bang Ye Wu, Kun-Mao Chao, "Spanning trees and optimization problems", *Chapman&Hall/CRC*, 2004, pp.13-139.

[4] Tran Viet Chuong, Phan Tan Quoc, Ha Hai Nam, "Proposing heuristic algorithms to solve the Steiner Minimal Tree problem", *Proceedings of the 10th National Conference on Fundamental and Applied Information Technology (FAIR)*; The University of Danang, August 17-18, 2017, ISBN: 978-604-913-614-6, Pages 138-147.

[5] C. C. Ribeiro, M.C. Souza, "Tabu Search for the Steiner problem in graphs", *Networks*, 36, 2000, pp.138-146.

[6] Chin Lung Lu, Chuan Yi Tang, Richard Chia-Tung Lee, "The full Steiner tree problem", *Elsevier*, 2003, pp.55-67.

[7] Chen, Chi-Yeh, and Sun-Yuan Hsieh, "An efficient approximation algorithm for the Steiner tree problem", National Cheng Kung University, Taiwan, 2018.

[8] Davide Bilò (University of Sassari), "New algorithms for Steiner tree reoptimization", *ICALP*, 2018.

[9] Daniel Markus Rehfeldt, "Generic Approach to Solving the Steiner Tree Problem and Variants", Fachbereich Mathematik der Technischen University at Berlin, 2015.

[10] Ding-Zhu Du, J. M. Smith, J.H. Rubinstein, "Advances in Steiner trees", *Springer Science & Business Media 2000*, pp.1-322.

[11] Fichte Johannes K., Hecher Markus, and Schidler André, "Solving the Steiner Tree Problem with few Terminals", University of Potsdam, *Germany*, 2020.

[12] Ivana Ljubic, "Solving Steiner Trees – Recent Advances", *Challenges and Perspectives*, 2020.

[13] Jack Holby, "Variations on the Euclidean Steiner Tree Problem and Algorithms", St. Lawrence University, 2017.

[14] J.E.Beasley, OR-Library: URL http://people.brunel.ac.uk/ mastjjb/jeb/orlib/steininfo.html

[15] J. E. Beasley, "An SST-Based algorithm for the Steiner problem in graphs", *Networks*, 19, 1989, pp.1-16.

[16] Jeffrey H. Kingston, Nicholas Paul Sheppard, "On reductions for the Steiner problem in graphs", Basser Department of Computer Science the University of Sydney, Australia, 2006, pp.1-10.

[17] Jon William Van Laarhoven, "Exact and heuristic algorithms for the euclidean Steiner tree problem", University of Iowa, 2010, (Doctoral thesis).

[18] Job Kwakernaak, "Pipeline network optimization using Steiner nodes", Wageningen University and Research Centre, The Netherlands, 2020.

[19] L. Kou, G. Markowsky, L. Berman, "A Fast Algorithm for Steiner Trees", *acta informatica*, Vol.15, pp.141-145, 1981.

[20] M. Hauptmann, M. Karpinski (Eds), "A compendium on Steiner tree problems", 2013, pp.1-36.

[21] Martin Zachariasen, "Algorithms for Plane Steiner Tree Problems", University of Copenhagen, 1998 (PH.D. Thesis).

[22] Marcello Caleffi, Ian F. Akyildiz, Luigi Paura, "On the solution of the Steiner tree np-hard problem via physarum bionetwork", *IEEE*, 2015, pp.1092-1106.

[23] Matjaz Kovse, "Vertex decomposition of steiner wiener index and Steiner betweenness centrality", University of Wroc law, Poland, 2016.

[24] Nguyen Viet Huy, Nguyen Duc Nghia, "Solving graphical Steiner tree problem using parallel genetic algorithm", *RIVF*, 2008.

[25] Ondra Suchý, "Exact algorithms for Steiner tree", Faculty of Information Technology, Czech Technical University in Prague, Prague, Czech Republic, 2014.

[26] Pieter Oloff De Wet, "Geometric steiner minimal trees", university of South Africa, 2008 (Thesis).

[27] Prosenjit Bose, Anthony D'Angelo, Stephane Durocher, "On the Restricted 1-Steiner Tree Problem", *Funded in part by the Natural Sciences and Engineering Research Council of Canada*, 2020.

[28] Prof. James Orlin, 15.082J Network Optimization MIT Course, Fall 2010.

[29] Radek Hušek, Dušan Knop, Tomáš Masarík, "Approximation Algorithms for Steiner Tree Based on Star Contractions: A Unified View", *International Symposium on Parameterized and Exact Computation (IPEC 2020)*, ACM, 2020.

[30] Reyan Ahmed and et al, "Multi-level Steiner trees", *ACM J. Exp. Algor.*, 1(1), 2018.

[31] Robert Sedgewick, Kevin Wayne, "Algorithms", Fourth edition, Addsion-Wesley, pp.518-700, 2011.

[32] S. E. Dreyfus, R. A. Wagner, "The Steiner Problem in Graphs", *Networks*, Vol.1, pp.195-207, 1971.

[33] Siavash Vahdati Daneshmand, "Algorithmic Approaches to the Steiner Problem in Networks", *Mannheim*, 2003.

[34] Thomas Pajor, Eduardo Uchoa, Renato F. Werneck, "A robust and scalable algorithm for the Steiner problem in graphs", *Springer*, 2018.

[35] Thomas, Bosman, "A solution merging heuristic for the Steiner problem in graphs using tree decompositions", *International Symposium on Experimental Algorithms*, Springer 2015, pp.1-12, 2015.

[36] Thorsten Koch, Alexander Martin, "Solving Steiner tree problems in graphs to optimality", Germany, 1996, pp.1-31.

[37] Tobias Polzin, "Algorithms for the Steiner Problem in Networks", Universit¨at des Saarlandes, 2003 (Thesis).

[38] Vu Dinh Hoa, "Steiner Tree Problem", *Institute of Mathematics*, Vietnam Academy of Science and Technology, http://math.ac.vn.

[39] Wassim Jaziri (Edited). "Local Search Techniques: Focus on Tabu Search". In-teh, Vienna, Austria, 2008, pp.1-201.

[40] Xinhui Wang, "Exact algorithms for the Steiner tree problem", *doctoral thesis*, ISSN 1381-3617, 2008.

[41] Xiuzhen Cheng, Ding-Zhu Du, "Steiner trees in industry", *Kluwer Academic Publishers*, vol.5, pp.193-216, 2004.

[42] Xin-She Yang, "Engineering optimization", *WILEY*, 2010, pp.21-137.

[43] Xin-She Yang, "Nature-inspired metaheuristic algorithms", *LUNIVER Press*, 2010, pp.53–62.

[44] Yahui Sun, "Solving the Steiner Tree Problem in Graphs using Physarum-inspired Algorithms", *arXiv preprint arXiv:1903.08926*, 2019.

[45] Zhiliu Zhang, "Rectilinear Steiner Tree Construction", Lincoln, Nebraska, USA, 2016.

Chuong Tran-Viet received a Master of Science degree in Computer Science in 2005 from Hanoi National University of Education.
He is currently working at the Center for Information Technology and Communications of Ca Mau province, Department of Information and Communications of Ca Mau province, Viet Nam.
Research Area: Evolutionary and Optimization Algorithms.
Email: chuongtv@camau.gov.vn

Quoc Phan-Tan received Master degree of computer science in 2002 from University of Science, Vietnam National University Ho Chi Minh City and received Doctor degree of computer science in 2015 from Hanoi University of Science and Technology (HUST).
He is currently a lecturer of Information Technology Faculty of Sai Gon University, Ho Chi Minh City, Viet Nam.
His research interests are metaheuristic algorithms.
Email: quocpt@sgu.edu.vn

Nam Ha-Hai received a Master of Informatics degree in 2004 from Hanoi University of Science and Technology and Ph.D in 2008 in UK.
He is currently working at: Vietnam National Institute of Software and Digital Content Industry, Ministry of Information and Communication, Viet Nam.
Research Area: Distributed systems and optimization.
Email: namhh@ptit.edu.vn

A Survey on Medical Image and Its Segmentation

Thuy Le-Nhi-Lam[1,2], Sang Vu-Ngoc-Thanh[1], Hieu Huynh-Trung[2], Bao Pham-The[1]

[1] Information Science Faculty, Sai Gon University, Vietnam
[2] Information Technology Faculty, Industrial University of Ho Chi Minh City, Vietnam
Correspondence should be addressed to Pham The Bao: ptbao@sgu.edu.vn

Communication: received 1 October 2021, revised 26 January 2022, accepted 28 February 2022
DOI: 10.32913/mic-ict-research.v2022.n1.1002

***Abstract*: Medical image processing and analysis are critical for assisting physicians with non-invasive clinical diagnoses. Understanding medical imaging techniques enables you to tackle corresponding medical image processing methodologically. Each imaging technique has distinct properties and produces images of various body regions. Medical image segmentation is crucial to improving treatment accuracy and assisting doctors' diagnosis conclusion. This article discusses medical imaging techniques and approaches to medical image segmentation.**

Keywords: *Medical images, medical image processing, medical image segmentation.*

I. Introduction

Medical images are created using specialized techniques and processes to depict information about the human body (or specific parts of the body) for various clinical purposes, including medical procedures, diagnostics, medical science. It is a subset of bioimaging and address radiology, endoscopy, thermography, medical imaging, and microscopy in a larger sense. For example, recording techniques such as electroencephalography (EEG) and magnetoencephalography (MEG) are designed to measure functional and effective brain connectivity. Applied studies and interpretation of medical images frequently focus on assessing pathology in the medical sciences (neuroscience, cardiology, psychiatry, and psychology) using comparable medical images. Numerous medical imaging systems have been created for scientific purposes and recently expanded to industry. Although mathematics was widely utilized for image processing, it had little impact on the biomedical area until the introduction of computed tomography (CT) for radiology, resulting in computer-aided tomography (CAT) procedures. Subsequently, isotope-emission tomography was created, allowing the development of positron emission tomography (PET) scan and single-photon emission tomography (SPECT) scan. Magnetic Resonance Imaging (MRI) has grown in popularity in recent decades, outpacing other imaging techniques [1].

Numerous studies have been researched for ultrasound, EEG applications, and novel optical imaging techniques, apart from these well-established approaches. While medical images are similar in appearance, the technology utilized to create them and their parameters are relatively different. This distinction is described in their features and usefulness in medicine. Numerous approaches have been developed to enable the physician to generate 3D images from CT, MRI, and ultrasound scanning software. Typically, a single CT or MRI scan results in a two-dimensional image on film. Repetition of the preceding steps results in creating a three-dimensional image [2].

Medical imaging is a subset of biological imaging, which has been developed since the 19th century. The following is a brief summary of medical imaging:

- In 1895, Roentgen accidentally discovered X-rays [3]. Conventional radiography is by far the most common medical imaging technique in science. Radionuclides have been employed for therapeutic and metabolic tracer research rather than imaging since 1896, leading to the invention of the linear γ-ray image scanner.
- Sonar technology was employed during World War II, and in 1970, ultrasonography became widely available in medicine. A brief history of ultrasound is mentioned explicitly in another document [4].
- In the twentieth century, mathematical expertise for tomographic reconstruction was established, as well as positron emission tomography (PET) and computed tomography (CT) techniques [5]. In magnetic resonance imaging, nuclear magnetic resonance has been employed for imaging (MRI).
- New types of X-ray, MRI, and ultrasound images are being produced in the twenty-first century, particularly those that comprise microscopy and macro-

scopic biological structures (thermal imaging, electrical impedance tomography, and probe scanning technique).

Medical images assist doctors in gaining additional information about a patient to make a more accurate diagnosis. Physicians appraise their patients based on their skills and experience. However, as the number of patients increases, the examination puts the doctor under increased pressure, and errors might arise. As a result, medical image segmentation enables physicians to save time on manual image analysis while minimizing potential errors. Physicians can save time when they employ scientific and technical solutions to patients for other medical duties. Medical images range from the simplest, such as a chest X-ray, to the most complicated, such as functional magnetic resonance imaging (fMRI). We have witnessed the phenomenal growth of new, powerful technology for detecting, storing, transmitting, analyzing, and displaying digital medical images over the last several decades. These techniques enable quantitative measurements to be made by biochemists, biologists, medical scientists, and physicists, thereby facilitating the validation of scientific hypotheses and medical diagnoses. The gathering of functional and metabolic data and structural data from medical images will be enhanced in the future. This approach can be accomplished using photon emission tomography (PET) and magnetic resonance spectroscopy [6]. Although computer-based medical image processing has been developed remarkably, it is challenging for computers to replace doctors completely. Therefore, the current solution is to develop technology to assist doctors in diagnosing and assessing patients' health. More precisely, medical image segmentation enables physicians to concentrate on areas of interest when making decisions. Additionally, image segmentation ensures the path for future technologies and increases the valuable content of diagnostic images.

II. Medical image processing model

Medical image analysis and interpretation are simplified using a three-level processing architecture, as seen in Figure 1. The system's first stage is image generation. Numerous imaging modes were discussed previously, including X-rays and SPECT. The image processing process is divided into low-level and high-level processing stages. Filtering, image augmentation, segmentation, and feature extraction are essential for further analysis. At this level of raw pixels processing, the most critical tasks are tumor identification and disease diagnosis.

The following five steps describe the fundamental procedures of image processing:

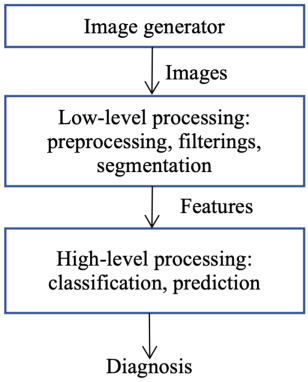


Figure 1. A model of medical imaging diagnostic system [7]

(i) Preprocessing aids in the visualization of low-resolution object boundaries. The most often used techniques are image registration, image transformation, filtering, and Laplace operators.

(ii) Filtering: Filtering includes enhancement, sharpening, and edge detection. Enhancement techniques include linear or nonlinear, local or global, or wavelet-based filters. Typical sharpening techniques are inversely or Weiner filters. Edge detection techniques include Haar transform, local operators, prediction, and/or classification methods.

(iii) Segmentation: Segmentation can be both region-based and boundary-based. There are different types of segmentation algorithms, including region-based classical algorithms, clustering algorithms, and line and arc detectors. An important issue in medical imaging is that it is possible to perform segmentation for various domains using bottom-up generalization methods without requiring special domain knowledge.

(iv) Shape modeling: Shape modeling is carried out using characteristics that can be employed independently or in conjunction with dimensional data. It is occasionally necessary to characterize the shape of an object in greater detail than is offered by a single but condensed characteristic reflected in the object image. In these cases, the shape descriptor expresses an object's shape in a more concise manner.

(v) Classification: Classification is based on feature selection, texture features, and decisions regarding feature classes. Each abnormality or disease is recognized as belonging to a specific group, and recognition is performed as a classification process.

III. Medical imaging techniques

Most medical imaging techniques are non-invasive. There are four types of medical images made from: (1) X-rays,

(2) γ-rays, (3) ultrasonic echoes, (4) induced magnetic resonance (MR), and (5) microscopic images. Normally images of types (3) and (4) are combined as described in Table1 I

TABLE I
MEDICAL IMAGE APPLICATION

Techniques	Inspected body parts/organs
X-ray	Chest, lungs, bones
γ-ray	Brain, internal organs, cardiac function
MR	Soft tissue, brain, fetus, pathological changes, internal organs
Microscopic images	Blood cells, bacteria

The conventional way of creating medical images is depicted in Figure 2. Ionizing radiation is seen in Figure 2.a and 2.b. Projection radiography and tomography are based on the passage of X-rays through the body and the selective attenuation of these rays by the body's tissues to generate images. The magnetic resonance effect, depicted in Figure 2.c, is caused by the nuclear magnetic resonance features. This indicates that protons gravitate toward the magnetic field. Selective stimulation of areas within the body can cause these protons to rotate in the opposite direction of the magnetic field, as illustrated in Figure 2.d, an ultrasound image created using high-frequency sound waves transmitted into the body and sound.

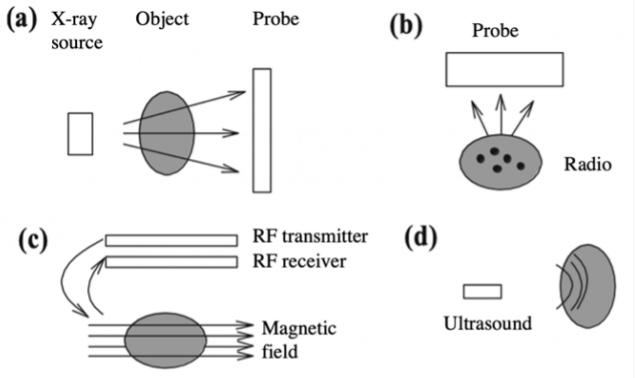


Figure 2. Medical imaging modalities: (a) X-ray images, (b) radionuclide images, (c) MR, and (d) ultrasound

1. Ionizing radiation images

W.C. Roentgen discovered X-ray in 1895, and it is the most extensively used type of medical imaging. For medical imaging, X-rays are a type of ionizing radiation with an energy range of 25 keV to 500 keV. An X-ray tube in a traditional radiography system produces a short pulse of X-rays that go through the human body. Non-absorbed or scattered X-ray photons reach the large area detector, which forms an image on the film, as illustrated in Figure 3. Energy loss as a function of the linear attenuation coefficient distribution in the body using a spatial model, Equation (1) captures this matter and energy dependent impact.

$$I_d = \int_0^{E_{max}} S_0(E) E e^{[-\int_0^d \mu(s;E)ds)]} dE \qquad (1)$$

where $S_0(E)$ is the X-ray spectrum and $\mu(s;E)$ is a function that describes the linear attenuation coefficient along the line between the source and the detector. S is the distance from the origin and d is the distance from the source to the detector. Image quality is affected by both noise resulting from the random nature of the X-rays or the transmission process as in Figure 3. Computed

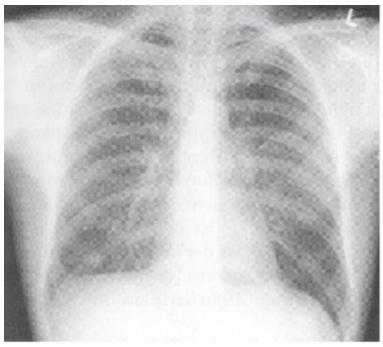


Figure 3. Chest X-ray image

tomography (CT), established by Hounsfield in 1972, is another common imaging technique that eliminates the effects of tissue overlap that obstruct an accurate diagnosis. CT collects X-ray projections around the patient, which can be seen as a series of X-rays obtained as the patient rotates gently around an axis. The films depict 2-D projections of the 3-D body from various viewpoints. A transverse line in the video represents a 1-D perspective of a 2-D axial cross-section of the body, represented by a collection of horizontal lines at the same height. The Radon transform [8] was used to reconstruct two-dimensional cross-sectional slices of the object using projection data. The Radon transform [8] is an integral transformation established by J. Radon in 1917. This transformation gathers 1-D projections of a 2-D object from various angles and reconstructs them using back propagation. The reconstruction is based on the Fourier slice theorem, which claims that a projection's 1-D Fourier transform is a slice of the object's 2-D Fourier transform as presented in Figure 4.

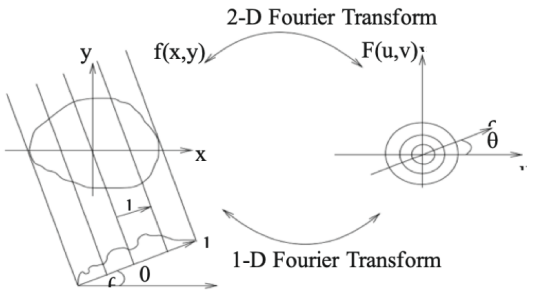


Figure 4. Visualize for projection-slice theorem

The basic imaging equation is similar to traditional radiography, except for a set of projections employed in cross-sectional image reconstruction as Equation (2).

$$I_d = I_0 e^{[-\int_0^d \mu(s;\bar{E})ds)]} dE \tag{2}$$

where, I_0 is the standard intensity, and $\bar{(E)}$ is the effective energy is E. CT has several advantages over projection radiography: (1) it eliminates the superimposition of images of structures outside the area of interest, (2) it provides high contrast resolution, allowing differences to be observed, and (3) as a tomographic and potentially three-dimensional approach for analyzing isolated cross-sectional visual slices of the body, to distinguish between tissues with less than 1% physical density. CT scans are a valuable complement to medical imaging since they provide more information than X-rays or ultrasound. Figure 5 5 shows how CT is used to diagnose cerebrovascular illnesses, acute and chronic alterations in the lung parenchyma, and the comprehensive diagnosis of the abdominal and pelvic organs. Nuclear medicine was first practiced in the late 1930s, and several operations involved radioactive materials. The use of radioactive iodine for the treatment of thyroid illness is known as initiation. Nuclear medicine imaging, like X-ray imaging, has progressed from projection to tomographic imaging. Nuclear medicine uses ionizing radiation and imaging techniques similar to radiography, but it focuses on physiological function rather than anatomy. Radiotherapy equipment, on the other hand, is used in nuclear medicine to receive rays from a radioactive source that is injected into the body. Distinct radiations perform different tasks and provide different information in nuclear medicine. In other words, different radiography technologies can visualize a wide spectrum of physiological and metabolic activities. A camera (external imaging device) records the patient's radiation and converts it into a flat, 2-D, or cross-sectional images. Nuclear medicine is a branch of medicine that deals with clinical diagnosis and treatment. It has many applications, including tumor diagnosis and treatment, acute care, cardiovascular, neurological, gastrointestinal, and renal problems. The three primary imaging modalities in nuclear medicine are divided into two main areas based on radiopharmaceutical decay: (1) planar imaging and single-photon emission tomography (SPECT), which uses gamma-emitting as a radiotherapy machine, and (2) positron emission tomography (PET), which uses positrons as an X-ray machine. An anger imaging camera creates a projection image, also known as a planar scan. Equation (3) – the basic image equation – has two crucial components: functioning as the desired parameter and attenuation as an undesired but essential addition.

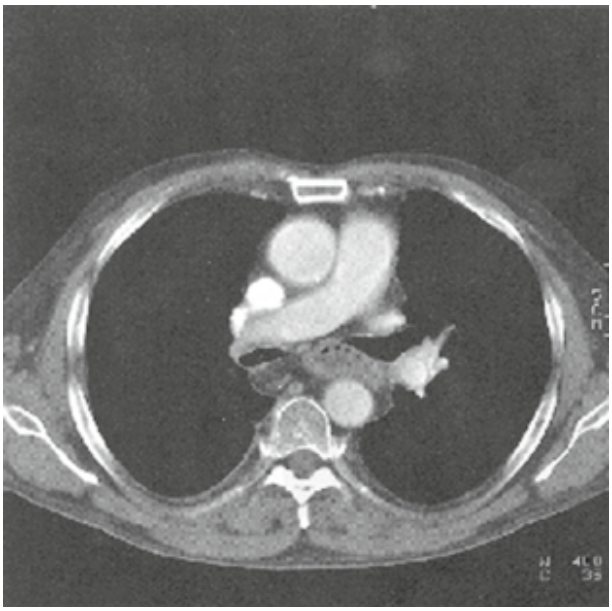


Figure 5. Visualization of the projection slice theorem.

$$\phi(x,y) = \int_{\infty}^{0} \frac{A(x,y,z)}{4\pi z^2} e^{[-\int_0^d \mu(x,y,z',E)dz']} dz \tag{3}$$

where A(x, y, z) represents the activity in the body and E is the energy of the photon. Image quality is largely determined by the resolution of the camera, and noise comes from system sensitivity, injected activity, and acquisition time.

On the other hand, SPECT collects detection data from several angles using an Anger camera. Single-photon emission is monitored with a gamma camera system and uses nuclei that decay by emitting a single photon. The acquired data is rebuilt using a image to form a 3-D dataset or thin (two-dimensional) slices in SPECT. This imaging technique can be thought of as a series of projections, each of which is a homogenous planar view. We must remember that if x and y are linear coordinates in the plane, then the plane line is expressed as Equation (4).

$$L(l,\theta) = \{(x,y)|x\cos\theta + y\sin\theta = l\} \tag{4}$$

where l is the lateral position of the line segment and θ is the angle of the one unit normal to the line segment, Figure 6. With the parameterize coordinates according to Equation (5), the integral f(x,y) is calculated according to the Equation (6.

$$\begin{cases} x(s) = l\cos\theta - s\sin\theta \\ y(s) = l\sin\theta + s\cos\theta \end{cases} \tag{5}$$

$$g(l,\theta) = \int_{-\infty}^{\infty} f(x(s), y(s))ds \tag{6}$$

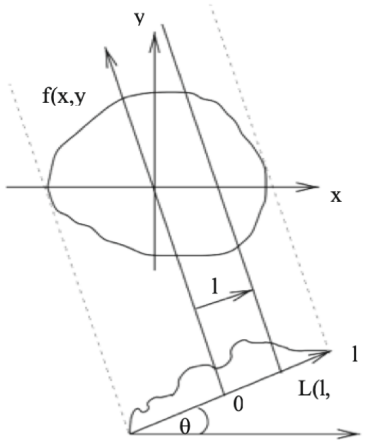


Figure 6. Geometric representation of lines and projections.

For a fixed angle θ, $g(l,\theta)$ is a projection, while for all l and θ, it is called a 2-D Radon transformation of $f(x,y)$. The image equation for SPECT is given as Equation (7).

$$\phi(l,\theta) = \int_{-\infty}^{\infty} A(x(s), y(s))ds \tag{7}$$

where $A(x(s), y(s))$ represents the inverse 2-D Radon transformation of (l,θ) and describes the radioactivity inside 3-D objects. As a result, there is no closed-form solution for attenuation correction in SPECT. SPECT is a significant imaging technology for obtaining functional images of organs because it provides accurate placement in 3-D space. Figure 7 shows a critical application in cardiac and brain function imaging.

PET is a special technique that has no similarities with other imaging modalities. PET radionuclides release positrons rather than x-rays. The positrons are the anti-electrons, and their locations are determined and measured. The reconstruction is built using the back-projection approach. PET imaging equations are identical to SPECT imaging equations with one exception: the integration limit for the attenuation component is extended over the entire

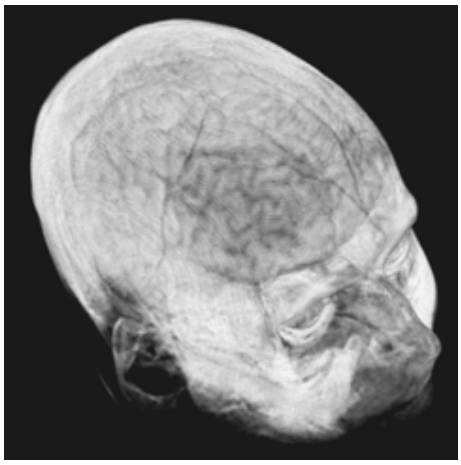


Figure 7. Brain imaging with SPECT.

body due to the simultaneous detection of paired γ-rays, referred to as the canceling photon as Equation (8).

$$\phi(l,\theta) = K\int_{-\infty}^{\infty} A(x(s), y(s))ds \tag{8}$$

where K is a constant that accounts for the influence of constant factors such as detector area and efficiency ϕ. Both SPECT and PET have image quality limitations due to resolution, scattering, and noise. PET is widely used in oncology, neurology, and psychiatry. A significant part focuses on neurological illnesses such as Alzheimer's disease, dementia, and epilepsy.

2. Magnetic resonance images

Magnetic resonance imaging (MRI) is a non-invasive imaging technique that produces images of the interior of the body. Over the last three decades, this technique has developed into a critical bio imaging modality in medicine. Nuclear medicine and MR imaging both provide information on pathological and physiological changes in bodily tissues, as well as structural details about organs. The magnetic resonance signal is generated by the nuclear magnetism of hydrogen atoms found in the body's fat and water and is based on the basic basis of nuclear magnetic resonance (NMR). NMR is associated with the electric charge and angular moment that particular nuclei possess. The nucleus possesses a positive charge and, when the atomic or mass number is odd, an angular momentum ϕ. Rotating these nuclei activates NMR. A tiny magnetic field should also be present around each revolving nucleus. When a magnetic field is applied externally, the direction of rotation tends to match the magnetic field. This is referred to as nuclear magnetism. Consider a particular rotating system (hydrogen atom) in a sample in the MR images. The term "sample" refers to a small piece of

tissue – a voxel. The rotation system becomes magnetized when a static magnetic field B_0 is applied and can be described using a bulk magnetization vector M. M will attain equilibrium M_0 in the undisturbed state, but parallel to the direction of B_0, Figure 9a. The critical point is that $M(r,t)$ is a function of time and that external radio frequency excitations and magnetic fields can be used to modify the three-dimensional coordinate r in space. The value of an MR image at a specific voxel is determined by two critical factors: tissue characteristics and the imaging methodology used by the scanner. The T1 and T2 relaxation parameters and the proton density are the most important tissue features. The term "proton density" refers to the number of target nuclei per unit volume. The scanner's software and hardware manage the temporal and spatial M magnetization vectors based on the pulse sequence. The equation of motion of $M(t)$ with respect to time t is based on the Bloch equation and the resonant frequency ω_0 or the Larmor frequency. $M(t)$ is composed of two components:

- The vertical magnetization is given by $M_z(t)$, with the z component of $M(t)$.
- $M_{xy}(t)$ is a complex quantity composed of two orthogonal components, Equation (9). The complex number M_{xy}, referred to as the phase angle ϕ, which is determined using the Equation (10).

$$M_{xy}(t) = M_x(t) + jM_y(t) \tag{9}$$

$$\phi = tan^{-1}\left(\frac{M_x}{M_y}\right) \tag{10}$$

A moment is generated when an external time-varying magnetic field $B(t)$ is applied because $M(t)$ is a magnetic moment. $B(t) = B_0$ if the field is static and parallel to the z-axis. It rotates if the magnetization vector M is initially pointed away from B_0. RF signals can also be used to stimulate rotating systems. This RF stimulation is accomplished by applying B_1 at the Larmor frequency rather than maintaining it constantly, so tracking the position of $M(t)$. This value, however, is not permanent, and we shall demonstrate two distinct ways for reducing motion and obliterating the received signal: longitudinal and transverse stretch. $M(t)$ is pushed downward by an angle with respect to the plane due to RF stimulation with B_1 is perpendicular to the y-axis. For $\alpha = 0$, we have $M_z + 0$ and the magnetization vector rotates in the xy-plane with a frequency equal to the Larmor frequency. The pulse B_1 required for an angle $\alpha = \pi/2$ is called pulse 90. The magnetization vector returns to its equilibrium and the dilation process is described by Equation (11) and will depend on the relaxation time according to vertically or in vertical rotation (T1), Figure 8.

$$M_z(t) = M_0\left[1 - e^{-\frac{t}{T_1}}\right] \tag{11}$$

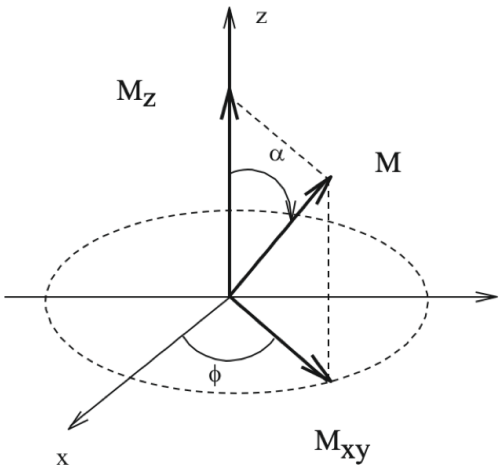


Figure 8. Magnetization vector M processed along the z-axis.

Horizontal stretching, also known as rotation-rotation, is the result of perturbations generated by other rotations when their phase is altered in relation to other rings. As a result of this attenuation, the reception antenna's signal is lost. The term "free inductive decay" refers to the received signal (FID). Equation describes the process of horizontal magnetization M_{xy} returning to equilibrium Equation (12).

$$M_{xy}(t) = M_{x_0y_0}e^{-\frac{t}{T_1}} \tag{12}$$

where T_2 is the rotational dilation time. T_2 is tissue-dependent and will produce contrast in the MR image. However, the received signal decays faster than T_2. Local noise in the static field B_0 gives rise to a faster time constant than T_2^* where $T_2^* < T_2$, Figure 9b. The decay process involving external field effects is modeled by the time constant T_2'. The relationship between the three horizontal expansion constants is modeled by Equation (13).

$$\frac{1}{T_2^*} = \frac{1}{T_2} + \frac{1}{T_2'} \tag{13}$$

Notably, T_1 and T_2 are tissue-dependent, and $T_2 \leq T_1$ for all materials. The time path of $\frac{T_1}{T_2}$ relaxation following the deployment of the RF pulse sequence provides critical information. The Fourier transform translates this measurable time process from the time domain to the frequency domain. The spectrum's amplitude corresponds to the resonant frequency of hydrogen nucleons in water, Figure 10. Contrast across tissues can be detected if the signals measured in those tissues are different. There are two ways to accomplish this: using intrinsic NMR properties such

as P_D, T_1 and T_2 or using attributes of externally delivered stimuli. Angle control is possible, as are sophisticated pulse sequences such as rotation-feedback. The 900-second pulse has a TR second repetition period followed by an 1800-second pulse after TE seconds. This second pulse provides an echo signal by replaying a partial rotation. The brain scan image is shown in Figure 11. The weights indicate that the observed difference in intensity between distinct tissues is primarily related to the tissues' P_D, T_1 and T_2 values. Contrast is created using the settings listed in Table II. MR images with pixel intensity $I(x, y)$ generated using the rotation-response series are defined by Equation (14).

$$I(x,y) \propto P_D(x,y)(1 - e^{-\frac{T_R}{T_1}})e^{-\frac{T_E}{T_2}} \tag{14}$$

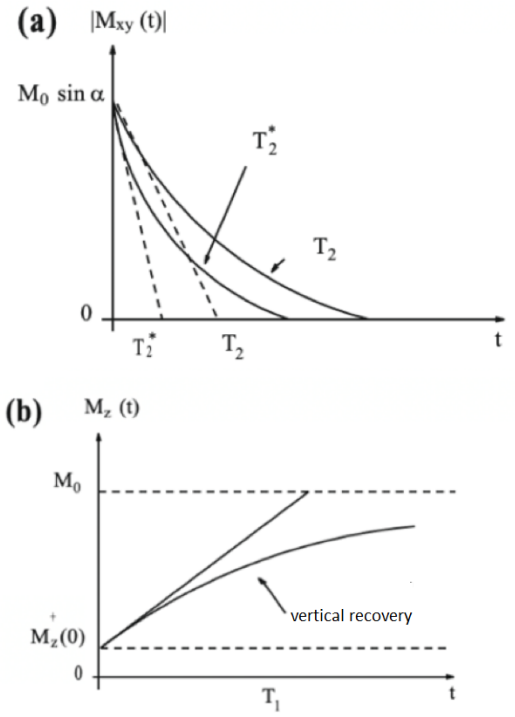


Figure 9. (a) Horizontal stretch and (b) Vertical stretch.

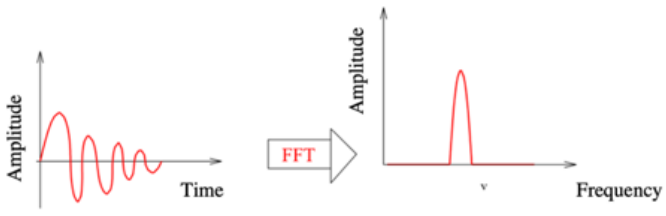


Figure 10. Frequency domain transformation of measured time process. Amplitude in the spectrum is represented at the Larmor frequency.

Adjusting the T_R and T_E values control the sensitivity of the signal to T_1/T_2 dilation and produce differently weighted

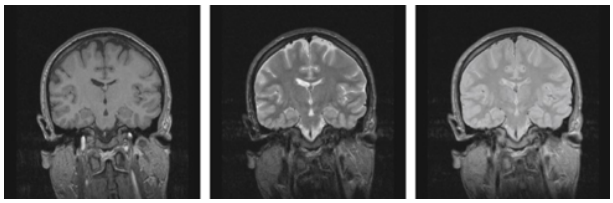


Figure 11. Brain MRI presetting (a) T_1, (b) T_2, and (c) hydrogen density weighted images

TABLE II
THE BASIC WAY TO CREATE CONTRAST DEPENDS ON T_1, T_2 AND P_D

Contrast	Scan parameters
P_D	Long T_R, FID, or short T_E
T_2	Lonng T_R, $T_E \approx T_2$
T_1	FID or short T_E, $T_R \approx T_1$,

contrast images. For instance, if T_R is significantly greater than T_1 for all tissues within the ROI, the T_1 weights converge to zero, indicating that signal sensitivity to T_1 dilation is absent. The same holds true for T_E, which is significantly less than T_2 in all tissues. When both the T_1 and T_2 sensitivities are decreased, the pixel density is solely determined by the proton density $P_D(x, y)$. The quality of an MR image is determined not only by contrast but also by sampling and noise. Magnetic resonance imaging with multiple dimensions is a critical method in MR. A sequence of three-dimensional magnetic resonance images with the same ROI was captured, provided that the images were registered correctly. This type of image enables the differentiation of various types of tissue. Contrast agents (CA) were utilized to correct the dilation time to further increase the contrast between tissue types. CA is injected intravenously, and throughout this period, signal increase for vascular-enhanced tissues occurs. Functional Magnetic Resonance Imaging (fMRI) is a novel non-invasive technique for examining the brain's cognitive activities [9]. The MR signal is vulnerable to changes in hemodynamic parameters such as blood flow, blood volume, and oxygen output during psychomotor activity, which is the basis for this approach. Blood Oxygenation Level-Dependent (BOLD) contrast is the most often used fMR signal. The BOLD time response changes when the local deoxyhemoglobin concentration falls in a neuronal active region, as shown in the T_2^* and T_2 weighted MR images.

The hemodynamic effect is defined by two fundamental characteristics: space and time. While the vascular system is primarily responsible for spatial effects, temporal effects account for the delay in detecting changes in the magnetic resonance signal in response to cerebral activity and

time. Hemodynamic changes disperse more slowly. Due to timing constraints, two distinct forms of fMR testing are required: "block" and "event-related" designs. The block designs are defined by an alternating sequence of 20–60 second blocks, including a test task. Multiple stimuli are provided at random, and the hemodynamic response to each stimulus is assessed in event-related designs. fMR with the excellent spatial and temporal resolution is an extremely powerful approach for visualizing the human brain's quick and fine activation processes as function localization is predicated on a solid connection between neuronal activity and MR signaling alterations. As is known from theoretical estimations and experimental results [10], the variability of the activated signal on clinical scans appears to be relatively low. This variation encourages analytical techniques to ascertain the response waveform and its related active regions. The primary advantages of this technique are that it allows for non-invasive recording of brain signals without the risk of radiation exposure associated with CT, (2) excellent spatial and temporal resolution, and (3) integration of fMR with other techniques such as MEG and EEG for studying the human brain. In pharmacology, fMR is a valuable technique for determining the brain's response to a specific medication. Apart from therapeutic uses, the value of fMR in comprehending neurological and mental illnesses and in optimizing diagnostics continues to grow.

3. Ultrasound images

Since the early 1950s, ultrasound has been a leading and extensively investigated imaging modality. It is a noninvasive way of imaging using 1–10 MHz vibrations passing through soft tissues and fluids. The method's cost-effectiveness and portability have made it immensely popular. It is diagnostically significant because it enables the "visualization" of pathological alterations in internal organs and blood vessels, which aids in the identification of cancer. Ultrasound imaging works on a fairly simple principle: when sound waves released by the transducer contact with tissue and blood, some of the unabsorbed energy returns to the transducer and is measured. The resulting "ultrasound signature" is formed when ultrasound radiation interacts with a variety of tissue types and is then used for diagnosis. Table 3 shows the speed of sound in tissue as a function of tissue type, temperature, and pressure (some examples of acoustic properties of some biological materials and tissues). Sound waves are "seen" as a result of scattering, absorption, or reflection. $A(x) = A_0 e^{-\alpha x}$, where A is a function of amplitude, A_0 is a constant, and α is the attenuation coefficient, and x is the distance. The amplitude and phase of the feedback signal contain critical information about the interaction and the type of interference medium. The fundamental image equation is the inverse pulse-echo equation, which describes the relationship between the excitation pulse, the transducer face, the object's reflectivity, and the received signal. In ultrasound, we have the following image modes:

- A-mode or amplitude-mode: the simplest model in which the envelope of the reflected pulse is shown against time. Primarily used in ophthalmology to establish the relative distances between various eye parts and to locate the midbrain or myocardial infarction, figure 12.
- B mode or light mode: created by scanning the transducer beam in a plane, figure 13, and is employed for both stationary and moving structures, such as the heart valve.
- M mode, or motion mode: displays the A-mode signal corresponding to repeated pulses in a distinct column of the 2D image; this mode is typically used in conjunction with an ECG to determine valve movement.

TABLE III
ACOUSTIC PROPERTIES OF SOME BIOLOGICAL MATERIALS AND TISSUES

Type	Speed of sound (m/s)	Impedance (106kg/m2s)	Level of decline (dB/cm at 1 MHz)
Air	344	0.0004	12
Water	1480	1.48	0.0025
Fat	1410	1.38	0.63
Tissue	1566	1.70	1.2 –3.3
Liver	1540	1.65	0.94
Bone	4080	7.80	20.0

The two basic techniques used to achieve better sensitivity of echoes along the dominant direction are:

- Beam: increase the sensitivity of the probe for a particular direction.
- Dynamic focus: increases the probe's sensitivity to a particular point in space at a specific time.

4. Microscopic Images

A microscope enables the physician to view images of cells invisible to the human eye. A microscope image was captured during observation with the aid of a microscope, as illustrated in Figure 14. Objects with a diameter of fewer than 75 m cannot be identified with the naked eye. Additionally, cells (10 m in diameter), bacteria (1 m), viruses (100 nm), molecules (2 nm), and atoms are frequently observed under a microscope (0.3 nm) [11].

Observing a image under a microscope entails several

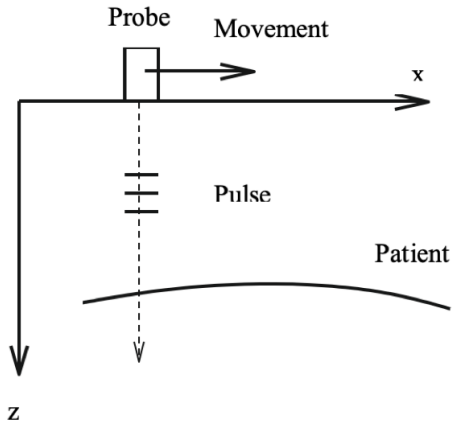


Figure 12. A mmode.

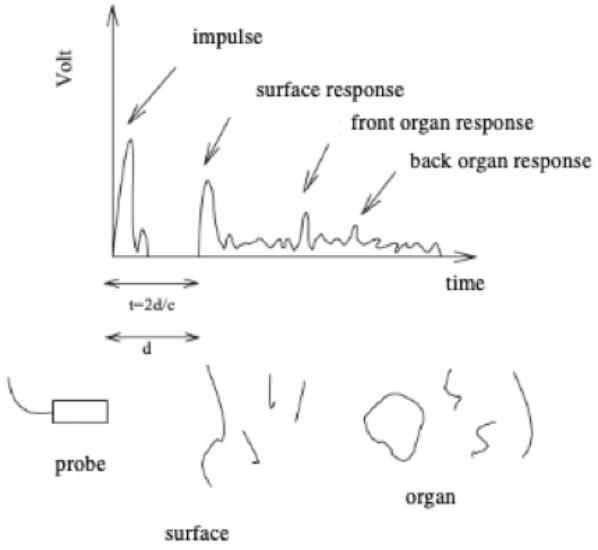


Figure 13. B mmode.

processes, including adjusting the light source, finding the specimen, and fine-tuning the focus. Locating significant cells is a critical step in the diagnostic use of microscopy. Chemical staining of red blood cells, white blood cells, platelets, and parasites facilitates microscopic examination.

IV. Medical image segmentation

Automating image processing and analysis procedures is critical for clinical diagnosis and treatment planning assistance. To identify regions of interest, consistent algorithms are necessary. (Regions of Interest – ROI) and anatomical structure. Computer-aided diagnosis is based on precise medical imaging analysis and utilizes information technology to produce rapid results. Segmentation is the process of breaking down a image into regions with similar

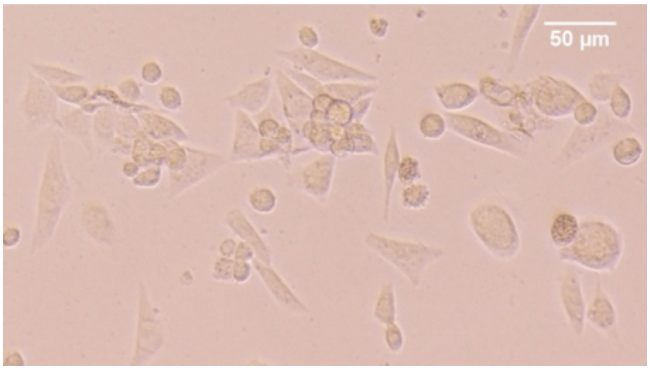


Figure 14. Cell culture medium.

attributes, such as color, gray level, contrast, brightness, and texture. The primary function of image segmentation in the medical industry is to define the region of interest (ROI), such as lesions, tumors, and any other anomalies, investigate the anatomical structure, quantify tissue volume, and aid in treatment planning. Automated medical image segmentation is challenging because of inhomogeneity, poor contrast, partial mass effect, artifacts, and relative gray level values of various soft tissues impacting segmented images. There is no uniform algorithm effective with all sorts of medical images, as each imaging technique has distinct advantages and disadvantages. Therefore, many segmentation approaches have been developed and utilized in medical applications, such as (i) shape-based segmentation (ii) interactive segmentation, and (iii) map-based segmentation. Segmentation techniques are further classified as follows: (i) histogram-based segmentation, region-based segmentation, and edge-based segmentation, all of which are based on gray-level features; and (ii) texture-based segmentation characteristics [12].

1. Histogram-based segmentation

Histogram-based image segmentation depends mainly on the threshold value of the histogram. This technique is used with uniformly bright areas in the image under consideration, where thresholding is used to segment the ROI and the background. The threshold can have one value or more values based on the number of ROIs in the image [13] or an appropriate threshold according to the characteristics of the image. Several algorithms are used to perform threshold segmentation efficiently. Typically, the entire image is scanned pixel by pixel to label the pixel as an object or background based on the gray level value compared to the threshold function (T), Algorithm 1.

Numerous improvements have been made to the fundamental histogram-based segmentation method, including

Algorithm 1: Segmentation Algorithm based on Global threshold

Inputs: Gray Image I and threshold T (from histogram of I)
Outputs: foreground R_1 and a set of and background R_2
begin
  for each pixel p in the image **do**
    if $p > T$ **then**
      p is assigned to set R_1;
    else
      p is assigned to set R_2
    end
    ;
  end
  Calculate $mean_1$ and $mean_2$ of R_1 and R_2;
  Calculate new threshold $T_{new} = \frac{mean_1+mean_2}{2}$;
  Repeat the previous step until the difference in T is less than the predefined threshold T_0 in subsequent iterations;
end

the following: adaptive histogram [14], local histogram [15], Kapur/Otsu [16], Gaussian mixture model [17].

2. Region-based segmentation

Segmentation is the process of splitting an image into distinct sections in order to ascertain the image's discrete regions directly [18], Algorithm 2.

Algorithm 2: Region-based segmentation algorithm (Region growing algorithm)

Inputs: Gray Image I and value of similarity
Outputs: object's regions
begin
  Iterative merge of a small region initially set based on the similarity;
  Select an arbitrary pixel;
  Compare that pixel with its neighboring pixels;
  Add similar neighbor pixels to increase the area size from the original pixel;
  if *The growth rate of an area stops* **then**
    Select another original pixel that does not belong to any region;
  end
  Repeat the whole process until all the pixels fit several areas;
end

Subsequently, numerous studies have been conducted to produce region-increasing algorithms with a method for selecting initial locations that maximize efficiency, such as evolving in the direction of regional spacing, such as k-means [19], fuzzy c-means [20], Level Set [21], Fast Marching [22].

3. Split and merge segmentation

Split and merge segmentation are based on the quartile tree data descriptor, which divides the image into four quadrants and provides heterogeneous root segmentation. Then, the squares' four neighbors are merged based on their segment identities [36]. This procedure of split/merge is repeated until no further splits/merges are possible, Algorithm 3.

Algorithm 3: Segmentation Algorithm based on Split and Merge regions

Inputs: Gray Image I, homogeneity condition and uniformity criteria for split region or merge regions
Outputs: object's regions
begin
  Define uniformity criteria;
  Divide the image into 4 quadrants;
  if *the quadrant is not uniform* **then**
    Divide the quadrant into 4 smaller quadrants
  end
  Merge two or more neighbors that satisfy the homogeneity condition at each level;
  Continue to split/merge until no more regions can be split or merged;
end

4. Edge-based segmentation

Edge-based segmentation is a powerful segmentation technique because edges contain a wealth of information about the image. Edges denote the boundary between two dissimilar regions; they have information about objects' placement, size/shape, and texture [23]. As a result, gradients can determine pixel values that differ between locations where the image intensity shifts from high to low or vice versa [24]. Different methods of edge-based segmentation include the Hough transform, contour detection, and edge stretching [37], Algorithm 4.

Algorithm 4: Edge-based segmentation algorithm

Inputs: Gray Image I and threshold T
Outputs: object's regions
begin
  Taking the derivative of the image to detect edges;
  Measure gradient amplitude calculate edge strength;
  Preserve all edges with magnitude greater than threshold T;
  Determine the dashed edges;
  Repeat the previous two steps with different threshold values to get closed margins;
end

5. Graph-based segmentation

Recently, a method for segmenting graph images using global functions has been proposed that produces better

TABLE IV
STUDIES APPLIED CONVENTIONAL SEGMENTATION METHODS

Ref	Medical images	Image type	Segmentation method	Segmentation algorithm	Resut
[13]	(1) Blood cell (2) Bacteria (3) Medium	Microscopic	Histogram	Threshold	(1) 120 (2) 122 (3) 125
[14]	(1) Retina (2) Knee joint (3) Brain (4) Breast (5) Uterus	(1) X-ray (2) X-ray (3) MRI (4) X-ray (5) X-ray	Histogram	Histogram equalization Cumulative histogram equalization Quadrant dynamic histogram equalization Contrast limited adaptive histogram equalization	MSE (1) 223 (2) 280 (3) 670 (4) 902 (5) 765
[16]	Chest	CT	Histogram	Kapur/Otsu	Accuracy: 97.62 %
[18]	Brain	MRI	Region-based	Probabilistic active contour	DICE 90 %
[19]	Brain	MRI	Region-based	K-mean clustering	Accuracy: 99.80 %
[23]	Heart	MRI	Edge-based	Threshold	Proposed method and parameters for segmentation
[24]	(1) Brain (2) Chest (3) Chest	(1) CT (2) MRI (3) CT	Edge-based	Active contour	Proof-of-concept
[25]	Chest	X-Ray	Graph-based	Minimum spanning tree	Sensitivity: 88.89 % Specificity: 95.83 %
[26]	Retina	Optical tomography	Graph-based	Shorted path	Error: Mean 2.17 Standard deviation 1.25 Median 1.53
[27]	Brain	MRI	Graph-based	N-cuts	Qualitative comparison with related studies

final segmentation results and fits the criteria of a wide variety of visual applications [38], [39], Algorithm 5.

Algorithm 5: Segmenting image by N-cuts

Inputs: Gray Image I
Outputs: object's regions
begin
  Convert medical image to graph $G = (V, E)$, calculate edge weight and matrix D and W;
  Solve $(D - W)y = \lambda Dy$ for the second smallest eigenvalue;
  Use eigenvectors to create eigenvalues and separate the graphs into 2 subgraphs;
  Decide whether to split the current partition or not. Recursively partition the partitioned parts, if necessary;
end

- The procedure for determining the graph's minimum spanning tree (Minimal Spanning Tree – MST): MST techniques for image segmentation describe the image by constructing a weighted graph against the feature space [25]. The grouping of pixels treated as a search for the graph's minimum spanning tree displays the image corresponding to the input image. By eliminating edges from the graph, subgraphs are created with a minimum sum of weights.
- Random walker: a proposed approach for performing interactive image segmentation with multiple labels. Utilizes a small number of user-defined labeled pixels to assess and quickly estimate the likelihood that a random walk beginning at each unlabeled pixel will end up at one of the labeled pixels. By labeling each pixel and calculating the maximum probability, high-quality image segmentation can be achieved. This technique has been used to segment the spinal cord using magnetic resonance imaging (MRI) images [40].
- Graph-cut technique using value functions [41]: similar to MST, this method also defines graphs using weighted graphs and global graph partitioning. The graph-cut method can be thought of as a template

TABLE V
STUDIES APPLIED MODERN METHODS FOR SEGMENTATION

Ref	Medical images	Image type	Segmentation method	Segmentation algorithm	Resut
[28]	Brain	MRI	Metaheuristics	Particle Swarm Optimization	Accuracy: 92.3 %
[29]	Brain	MRI	Metaheuristics	Shuffled Frog Leaping	DICE: 0.8652
[30]	Brain	MRI	CNN	2D CNN	DICE: 0.8503 ± 0.0227
[31]	Chest	X-ray	CNN	2D CNN	AUC: 0.93
[32]	(1) Brain (2) Chest (3) Heart	(1) MRI (2) MRI (3) CT	CNN	2.5D CNN	DICE: (1) 0.7 – 0.9 (2) 0.75 (3) 0.6
[33]	Brain	MRI	CNN	3D CNN	DICE: 0.84
[34]	Brain	MRI	CNN	3D CNN	DICE: 0.66
[35]	Brain	MRI	LSTM	U-net	DICE: 0.73

for image segmentation applications that use graph partitioning techniques. It is advantageous for various applications since it enables the specification of distinct 'cut' criteria and the optimization of global value computation functions for graph partitioning. Typical examples include the max-flow/min-cut algorithm and the Markov random field model. Typical graph-cut methods include the minimal cut, N-cuts, mean cut, and ratio cut [42], [43].

- The method is based on the object's or image's boundary to determine the shortest path through the graph [26]. Using standard graph theory techniques such as Dijkstra, the edge set is the shortest path between two vertices of a graph. Calculating the border of an object or partition in an image is equivalent to computing the shortest path between two graph vertices. Additionally, other applications demand user participation, such as selecting the boundary's starting point.
- N-cut problems [27].
  - Let $d(i) = \sum_i w_{ij}$
  - D is $n \times n$ diagonal matrix with the diagonal d.
  - Let W is the $n \times n$ symmetric matrix and $w_{ij} = w_{ji}$
  - Find the minimal $\min_{(S,\bar{S})} ncut(S,\bar{S}) = \min_y \frac{y^T(D-W)y}{y^T D y}$ where $y_i \in \{1, -b\}$ and $y^t D y = 0$
  - Minimize $\frac{y^T(D-W)y}{y^T D y}$ with the constraint y, the problem becomes finding the second smallest eigenvalue of the problem $(D - W)y = \lambda D y$

z

6. Machine Learning

Numerous studies have investigated the segmentation of various organs to extract problematic areas from medical images [44]. Active Appearance Models (AAM), Support Vector Machine (SVM), and Artificial Neural Network (ANN) are three supervised learning methods that are used to interpret medical images. The use of nonlinear statistics in ANN and SVM demonstrates complex modeling relationships between inputs and outputs. The classifier's weight is determined by maximizing organ, structural, and cell features by maximizing the differentiation function. These weights are redistributed after each sample in the training set is processed. The extracted information from the training set gives basic structural information, such as shape, location, and intensity, that can be evaluated as corresponding information for image segmentation. On the other hand, AAMs are statistical models of a structure's shape, with training samples used to extract shape parameter ranges, mean shape, and mean shape. To ensure comparability between segmentation results and training samples, conformational parameters should be restricted in areas where the segmentation technique is used to find the higher position based on appearance information of shape points. Thus, the classifier's algorithms can be widely applied to images of internal organs such as the brain and heart in medical imaging. Metaheuristic-based magnetic resonance imaging: Brain tumors originate when the cell shape within the brain becomes aberrant; they are classified into two forms, benign tumors, and malignant

tumors. Primary malignant tumors are those that have not metastasized elsewhere. Secondary malignant tumors are those that have metastasized elsewhere. MR imaging is one of the most advanced tools available for diagnosing brain tumors today. Additionally, segmentation is critical for extracting suspicious regions from complicated medical imaging of the brain. Automated brain tumor diagnosis through MR imaging represents an exciting promise for early and more accurate detection of brain malignancies. Gopal and Karnan [28] developed an intelligent method for diagnosing brain malignancies by combining PSO (Particle Swarm Optimization) and GA in the clustering phase of MR image processing (Genetic Algorithms). Ladgham et al. [45] developed a novel metaheuristic method dubbed SFLA (Shuffled Frog Leaping Algorithm) or modified MSFLA for brain MR image segmentation. MSFLA enables segmentation without the use of attenuating filters or with the use of three-dimensional Otsu.

7. Deep learning

- CNN: CNN [45] is a subtype of neural network that consists of a stack of layers, each of which performs a distinct action, such as convolution, aggregation, or loss computation. Each intermediate layer receives the output of the preceding layer as input, as seen in Figure 15. The first layer is an input layer connected directly to the input image. The following collection of layers is composite classes that produce the outcome of converting the input data through a specified number of filters and operate as an automatic feature extraction tool. Filters are frequently referred to as multipliers and have a size that is based on the data. Each convolutional layer's output is handled as an activation map, emphasizing the effect of a particular filter applied to the input data. The structure described above is referred to as the conventional CNN.
  - 2D CNN: With CNN's promising image classification and pattern recognition skills. The general concept pertaining to medical images is segmentation and applying 2D filters to the 2D input image. Zhang [30] improves segmentation results by combining numerous 2D images with distinct color channels (e.g., R, G, B). The results indicated that a particular type was more productive. Bar et al. [31] merged low-level characteristics from Imagenet with transfer learning. Originate from PiCoDes [46].
  - 2.5D CNN: 2.5D Approaches [32], [47], [48], The authors in [47] applied this idea to knee cartilage segmentation. Moeskops et al. evaluated fractional multitasking using the 2.5D architecture [32]. Whether a single network design is capable

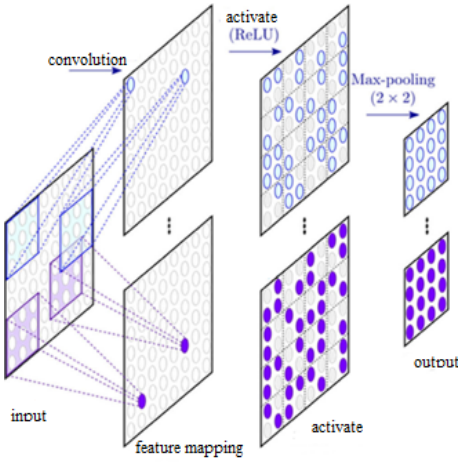


Figure 15. CNN structure.

of segmenting many organs, the author extended the concept even further by incorporating several modalities (for brain MR, breast MR, and cardiac CTA). 2.5D techniques make use of 2D labeled data, which is more accessible than 3D data and produces findings that are more consistent with existing technology. Additionally, breaking the cube into a random collection of 2D images alleviates the size issue [49].

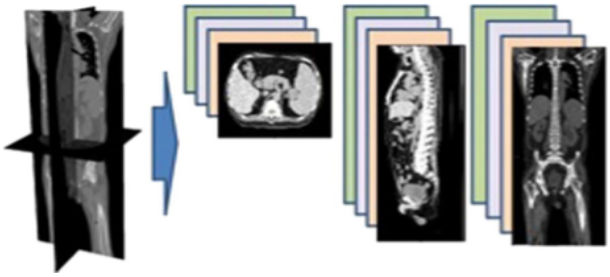


Figure 16. 2.5D example.

  - 3D CNN: Using a 2.5D framework enables the enrichment of spatial information. However, the 2.5D approaches remain restricted to the 2D kernel, preventing 3D filters. 3D CNNs are used to extract data along all three axes (X, Y, and Z). The network's construction is similar to that of a 2D CNN, except that 3D modules are used in each required part, for example, 3D composite layers and 3D subsampling layers. In one of the first pure 3D models, brain tumors of any size were segmented [33]. The proposal of Kamnitsas [34] is a bidirectional 3D CNN design; this enables the analysis of greater areas surrounding the voxel and enhances the entire system when dealing with multi-scale contexts. With

the reduced kernel size usage, this alteration 3×3 improved accuracy. Dou [50] offers to employ a collection of spatially shared 3D kernels to address the size and processing time issues. To segment an organ from complicated three-dimensional images. However, training such deep networks presents a substantial difficulty in 3D modeling. Three-dimensional compositing is used at the subsampling layer as feedback filtering in a small neighborhood to stabilize the learned features against local displacement in three-dimensional space. This enables a substantially faster convergence rate than pure 3D CNNs when a convolution mask with the same size as the input volume is used. Kleesiek [33] tackled the difficult task of brain boundary detection using 3D CNNs. The author applied binary segmentation via the cutoff function and mapped the outputs to the desired label, achieving a nearly 6 % improvement over other conventional methods.

  - Additionally, there are several enhancements, including : Fully Convolutional Network (FCN), Cascaded FCN (CFCN), Multi-Stream FCN, U-Net, 2D U-Net, 3D U-Net, V-Net, Convolutional Residual Networks (CRNs).

- Recurrent Neural Networks (RNNs): RNN Allows the network to retain knowledge from previous layers and use the preceding layer's output data as input data. Since the ROI in medical images is frequently scattered over multiple contiguous slices (e.g., in CT or MR), subsequent slices exhibit correlation. As a result, the RNN can extract adjacent slice contexts as sequential data from the input slices. RNN Structures are composed of two major components for the extraction of internal information that can be accomplished by any CNN model or RNN.
- LSTM: LSTM [51] is often regarded as the most well-known RNN model. A conventional LSTM network requires vectorization of the input, which is inconvenient for medical image segmentation because spatial information is lost. However, the excellent idea is to use convolutional LSTM (CLSTM) [52], [53], where vector multiplication is substituted with convolution. F. Xu and colleagues presented a brain tumor segmentation architecture using MRI images [35].
- Contextual LSTM (CLSTM): CLSTM is used to the output layer of the CNN in [54] to achieve crisper segmentation by utilizing context information from nearby slices. The approach significantly outperforms the well-known U-Net structure [55]. Chen [56] modified the CNN by adding a two-way CLSTM (BD-CLSTM) to the U-Net structure. Because BDC-LSTM can process data sequentially in two directions z+ and z- rather than only one, it outperforms tower LSTM [57] in terms of information acquired in six directions. (x+ , x- , y+ , y- , z+ , and z-). While the LSTM tower moves in six directions, combining the six outputs generated in each direction results in a loss of spatial information. As a result, BDC-LSTM should perform slightly better in the z direction.
- Gated Recurrent Unit (GRU): GRU is an LSTM version in which memory cells are omitted, and the structure becomes simpler without sacrificing speed [58]. Poudel [59] used GRU to analyze the FCN and regression FCN (RFCN) systems. The advantages of RFCN are that it can perform both detection and segmentation in a single build and can be trained for both FCN and GRU.
- Clockwork RNN (CW-RNN): The CW-RNN proposed in [60], [61] has promised long-term information modeling with fewer parameters than a pure RNN. This structure has been used to the portion of connective tissue that surrounds the perimuscular muscle [62], [62]. Since only one component of the CW-RNN is active at any given time, it is more efficient than other approaches (requiring 100 times the runtime of the modified CNN) [63], and a comparison of CW-RNN and U-Net demonstrates that, on average, a 5% increase in accuracy. It should be mentioned that parallelizing RNNs on a restricted number of GPUs is a complex operation, even more so when the data is three-dimensional [57]. Additionally, decoupling training for separate RNN modules increased the complexity and duration of the training process. In that case, RNN performs better with greater internal organs data and more interlayer information than with a small lesion section with the entire ROI captured in a single slice.

V. Challenges and Opportunities

1. Challenges

At the time of writing this article, no one ideal segmentation method exists for all types of medical images, such as coronary imaging [64], Figure 16. There are numerous causes for this, including the following:

- Sophisticated organ structures: The human body's blood arteries are complex. The coronary artery system is comprised of the right and left coronary arteries as well as several small blood vessels that surround the heart and are responsible for the circulatory system's pumping function. The number and position of these blood arteries within the coronary artery network

frequently differ between individuals. As a result, it is impossible to apply a model of one patient's vascular structure to another.

- Noise: is inevitability and the most concerning source of error in image processing. Depending on the type of imaging, whether X-ray or MR. Different strategies are required to deal with noise in different kinds of images.
- The contrast between topic and background: in some image areas, such as figure 18, the background will have the same gray level as the subjects. Depending on the time of the image, the volume of blood pushed up varies. Increased or decreased heart rate affects both the quality of blood vessels and the image.
- Non-uniform distribution of light: When capturing X-ray images, the light impact from the lamp and ambient light significantly affects the image quality, causing areas of the image to be bright and dark. However, in the absence of light, the items contained within the image will be unrecognizable. As a result, non-uniform light distribution is one of the constraints of X-ray image databases that must be recognized. Then, we should have strategies for resolving this circumstance.

2. Opportunities

Each medical category will provide information about a specific area of the body. The relevance study is based on a characteristic of each image kind and employs analytical relevancy. Medical problem analysis is one of the most complex problems; it is also critical for classification and subsequent analysis. The results of the problem segmentation, the medical image, will be used to determine the problem's level of accuracy later. With the issues described above and numerous more obstacles encountered during implementation, these are the formulas and an opportunity for many researchers to tackle the medical image analysis problem.

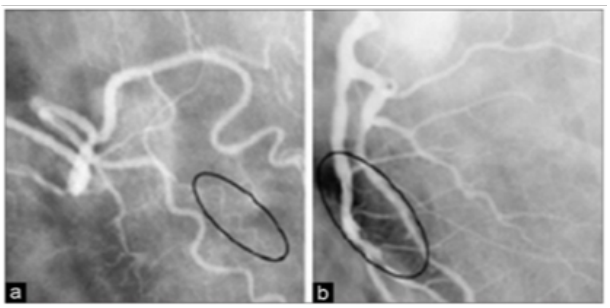


Figure 17. Limitations of coronary angiography: (a) low contrast, and (b) non-uniform light distribution.

References

[1] S. Webb, "The physics of medical imaging-medical science series," 1996.

[2] ScienceDirect Topics, "Medical imaging, medical imaging - an overview." [Online]. Available: https://www.sciencedirect.com/topics/computer-science/medical-imaging (accessed: 2022-01-01)

[3] History, "German scientist discovers x-ray." [Online]. Available: https://www.history.com/this-day-in-history/german-scientist-discovers-x-rays (accessed: 2022-01-01)

[4] Ultrasound schools info, "History of ultrasound." [Online]. Available: https://www.ultrasoundschoolsinfo.com/history/ (accessed: 2022-01-01)

[5] The evolution of imaging technology, "History of the pet/ct scanner." [Online]. Available: https://www.theevolutionofimagingtechnology.net/history-of-the-pet-ct-scanner/ (accessed: 2022-01-01)

[6] H. Zaidi, "Medical imaging: Current status and future perspectives," 1997.

[7] M. Akay, "Time frequency and wavelets in biomedical signal processing," 1998.

[8] Z.-H. Cho, J. P. Jones, and M. Singh, *Foundations of medical imaging*. Wiley New York, 1993, vol. 228.

[9] S. Ogawa, D. W. Tank, R. Menon, J. M. Ellermann, S. G. Kim, H. Merkle, and K. Ugurbil, "Intrinsic signal changes accompanying sensory stimulation: functional brain mapping with magnetic resonance imaging," *Proceedings of the National Academy of Sciences*, vol. 89, no. 13, pp. 5951–5955, 1992.

[10] S. Ogawa, T. Lee, and B. Barrere, "The sensitivity of magnetic resonance image signals of a rat brain to changes in the cerebral venous blood oxygenation," *Magnetic resonance in medicine*, vol. 29, no. 2, pp. 205–210, 1993.

[11] A. Maier, S. Steidl, V. Christlein, and J. Hornegger, "Medical imaging systems: An introductory guide," 2018.

[12] T. Heimann, B. Van Ginneken, M. A. Styner, Y. Arzhaeva, V. Aurich, C. Bauer, A. Beck, C. Becker, R. Beichel, G. Bekes *et al.*, "Comparison and evaluation of methods for liver segmentation from ct datasets," *IEEE transactions on medical imaging*, vol. 28, no. 8, pp. 1251–1265, 2009.

[13] O. J. Tobias and R. Seara, "Image segmentation by histogram thresholding using fuzzy sets," *IEEE transactions on Image Processing*, vol. 11, no. 12, pp. 1457–1465, 2002.

[14] N. Salem, H. Malik, and A. Shams, "Medical image enhancement based on histogram algorithms," *Procedia Computer Science*, vol. 163, pp. 300–311, 2019.

[15] S. Patel, K. Bharath, S. Balaji, and R. K. Muthu, "Comparative study on histogram equalization techniques for medical image enhancement," in *Soft Computing for Problem Solving*. Springer, 2020, pp. 657–669.

[16] S. C. Satapathy, D. J. Hemanth, S. Kadry, G. Manogaran, N. M. Hannon, and V. Rajinikanth, "Segmentation and evaluation of covid-19 lesion from ct scan slices-a study with kapur/otsu function and cuckoo search algorithm," 2020.

[17] F. Riaz, S. Rehman, M. Ajmal, R. Hafiz, A. Hassan, N. R. Aljohani, R. Nawaz, R. Young, and M. Coimbra, "Gaussian mixture model based probabilistic modeling of images for medical image segmentation," *IEEE Access*, vol. 8, pp. 16 846–16 856, 2020.

[18] E. R. Arce-Santana, A. R. Mejia-Rodriguez, E. Martinez-Peña, A. Alba, M. Mendez, E. Scalco, A. Mastropietro, and G. Rizzo, "A new probabilistic active contour region-based method for multiclass medical image segmentation," *Medical & biological engineering & computing*, vol. 57, no. 3, pp. 565–576, 2019.

[19] D. M. Kumar, D. Satyanarayana, and M. Prasad, "An improved gabor wavelet transform and rough k-means clus-

tering algorithm for mri brain tumor image segmentation," *Multimedia Tools and Applications*, vol. 80, no. 5, pp. 6939–6957, 2021.

[20] K. M. Aljebory and T. S. Mohammed, "Modified fuzzy c-means clustering algorithm application in medical image segmentation," *JEA Journal of Electrical Engineering*, vol. 2, no. 1, pp. 1–9, 2018.

[21] Z. Zhang and J. Song, "An adaptive fuzzy level set model with local spatial information for medical image segmentation and bias correction," *IEEE Access*, vol. 7, pp. 27 322–27 338, 2019.

[22] D. Jia and X. Zhuang, "Directional fast-marching and multi-model strategy to extract coronary artery centerlines," *Computers in Biology and Medicine*, vol. 108, pp. 67–77, 2019.

[23] A. Tsai, A. Yezzi, W. Wells, C. Tempany, D. Tucker, A. Fan, W. E. Grimson, and A. Willsky, "A shape-based approach to the segmentation of medical imagery using level sets," *IEEE transactions on medical imaging*, vol. 22, no. 2, pp. 137–154, 2003.

[24] R. Hemalatha, T. Thamizhvani, A. J. A. Dhivya, J. E. Joseph, B. Babu, and R. Chandrasekaran, "Active contour based segmentation techniques for medical image analysis," *Medical and Biological Image Analysis*, vol. 4, no. 17, p. 2, 2018.

[25] S. Nithya, S. Bhuvaneswari, and S. Senthil, "Robust minimal spanning tree using intuitionistic fuzzy c-means clustering algorithm for breast cancer detection," *Am. J. Neural Netw. Appl*, vol. 5, pp. 12–22, 2019.

[26] X. Liu, D. Liu, T. Fu, Z. Pan, W. Hu, and K. Zhang, "Shortest path with backtracking based automatic layer segmentation in pathological retinal optical coherence tomography images," *Multimedia Tools and Applications*, vol. 78, no. 12, pp. 15 817–15 838, 2019.

[27] N. Ghorpade and H. Bhapkar, "Evaluation and performance analysis of brain mri segmentation methods," *Int. J. Computer Sci. Eng*, vol. 6, no. 8, pp. 687–696, 2018.

[28] N. N. Gopal and M. Karnan, "Diagnose brain tumor through mri using image processing clustering algorithms such as fuzzy c means along with intelligent optimization techniques," in *2010 IEEE international conference on computational intelligence and computing research*. IEEE, 2010, pp. 1–4.

[29] A. Ladgham, F. Hamdaoui, A. Sakly, and A. Mtibaa, "Fast mr brain image segmentation based on modified shuffled frog leaping algorithm," *Signal, Image and Video Processing*, vol. 9, no. 5, pp. 1113–1120, 2015.

[30] W. Zhang, R. Li, H. Deng, L. Wang, W. Lin, S. Ji, and D. Shen, "Deep convolutional neural networks for multi-modality isointense infant brain image segmentation," *NeuroImage*, vol. 108, pp. 214–224, 2015.

[31] Y. Bar, I. Diamant, L. Wolf, and H. Greenspan, "Deep learning with non-medical training used for chest pathology identification," in *Medical Imaging 2015: Computer-Aided Diagnosis*, vol. 9414. International Society for Optics and Photonics, 2015, p. 94140V.

[32] P. Moeskops, J. M. Wolterink, B. H. van der Velden, K. G. Gilhuijs, T. Leiner, M. A. Viergever, and I. Išgum, "Deep learning for multi-task medical image segmentation in multiple modalities," in *International Conference on Medical Image Computing and Computer-Assisted Intervention*. Springer, 2016, pp. 478–486.

[33] G. Urban, M. Bendszus, F. Hamprecht, and J. Kleesiek, "Multi-modal brain tumor segmentation using deep convolutional neural networks," *MICCAI BraTS (brain tumor segmentation) challenge. Proceedings, winning contribution*, pp. 31–35, 2014.

[34] K. Kamnitsas, L. Chen, C. Ledig, D. Rueckert, and B. Glocker, "Multi-scale 3d convolutional neural networks for lesion segmentation in brain mri," *Ischemic stroke lesion segmentation*, vol. 13, p. 46, 2015.

[35] F. Xu, H. Ma, J. Sun, R. Wu, X. Liu, and Y. Kong, "Lstm multi-modal unet for brain tumor segmentation," in *2019 IEEE 4th International Conference on Image, Vision and Computing (ICIVC)*. IEEE, 2019, pp. 236–240.

[36] J. Freixenet, X. Munoz, D. Raba, J. Martí, and X. Cufí, "Yet another survey on image segmentation: Region and boundary information integration," in *European conference on computer vision*. Springer, 2002, pp. 408–422.

[37] R. Hapsari, M. Utoyo, R. Rulaningtyas, and H. Suprajitno, "Iris segmentation using hough transform method and fuzzy c-means method," in *Journal of Physics: Conference Series*, vol. 1477, no. 2. IOP Publishing, 2020, p. 022037.

[38] B. Peng, L. Zhang, and D. Zhang, "A survey of graph theoretical approaches to image segmentation," *Pattern recognition*, vol. 46, no. 3, pp. 1020–1038, 2013.

[39] P. F. Felzenszwalb and D. P. Huttenlocher, "Efficient graph-based image segmentation," *International journal of computer vision*, vol. 59, no. 2, pp. 167–181, 2004.

[40] D. Brindha and N. Nagarajan, "An efficient automatic segmentation of spinal cord in mri images using interactive random walker (rw) with artificial bee colony (abc) algorithm," *Multimedia Tools and Applications*, vol. 79, no. 5, pp. 3623–3644, 2020.

[41] J. Dogra, S. Jain, and M. Sood, "Analysis of graph cut technique for medical image segmentation," in *International Conference on Advanced Informatics for Computing Research*. Springer, 2019, pp. 451–463.

[42] S. Wang and J. M. Siskind, "Image segmentation with minimum mean cut," in *Proceedings Eighth IEEE International Conference on Computer Vision. ICCV 2001*, vol. 1. IEEE, 2001, pp. 517–524.

[43] S. Wang and J. M. Siskind, "Image segmentation with ratio cut," vol. 25, no. 6, pp. 675–690, 2003.

[44] D. L. Pham, C. Xu, and J. L. Prince, "Current methods in medical image segmentation," *Annual review of biomedical engineering*, vol. 2, no. 1, pp. 315–337, 2000.

[45] F. Commandeur, M. Goeller, J. Betancur, S. Cadet, M. Doris, X. Chen, D. S. Berman, P. J. Slomka, B. K. Tamarappoo, and D. Dey, "Deep learning for quantification of epicardial and thoracic adipose tissue from non-contrast ct," *IEEE transactions on medical imaging*, vol. 37, no. 8, pp. 1835–1846, 2018.

[46] A. Bergamo, L. Torresani, and A. Fitzgibbon, "Picodes: Learning a compact code for novel-category recognition," *Advances in neural information processing systems*, vol. 24, 2011.

[47] A. Prasoon, K. Petersen, C. Igel, F. Lauze, E. Dam, and M. Nielsen, "Deep feature learning for knee cartilage segmentation using a triplanar convolutional neural network," in *International conference on medical image computing and computer-assisted intervention*. Springer, 2013, pp. 246–253.

[48] H. R. Roth, L. Lu, A. Seff, K. M. Cherry, J. Hoffman, S. Wang, J. Liu, E. Turkbey, and R. M. Summers, "A new 2.5 d representation for lymph node detection using random sets of deep convolutional neural network observations," in *International conference on medical image computing and computer-assisted intervention*. Springer, 2014, pp. 520–527.

[49] R. Fakoor, F. Ladhak, A. Nazi, and M. Huber, "Using deep learning to enhance cancer diagnosis and classification," in *Proceedings of the international conference on machine learning*, vol. 28. ACM, New York, USA, 2013, pp. 3937–3949.

[50] Q. Dou, L. Yu, H. Chen, Y. Jin, X. Yang, J. Qin, and P.-A. Heng, "3d deeply supervised network for automated segmentation of volumetric medical images," *Medical image analysis*, vol. 41, pp. 40–54, 2017.
[51] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.
[52] N. Srivastava, E. Mansimov, and R. Salakhudinov, "Unsupervised learning of video representations using lstms," in *International conference on machine learning*. PMLR, 2015, pp. 843–852.
[53] X. Shi, Z. Chen, H. Wang, D.-Y. Yeung, W.-K. Wong, and W.-c. Woo, "Convolutional lstm network: A machine learning approach for precipitation nowcasting," *Advances in neural information processing systems*, vol. 28, 2015.
[54] J. Cai, L. Lu, Y. Xie, F. Xing, and L. Yang, "Improving deep pancreas segmentation in ct and mri images via recurrent neural contextual learning and direct loss function," *arXiv preprint arXiv:1707.04912*, 2017.
[55] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," in *International Conference on Medical image computing and computer-assisted intervention*. Springer, 2015, pp. 234–241.
[56] J. Chen, L. Yang, Y. Zhang, M. Alber, and D. Z. Chen, "Combining fully convolutional and recurrent neural networks for 3d biomedical image segmentation," *Advances in neural information processing systems*, vol. 29, 2016.
[57] M. F. Stollenga, W. Byeon, M. Liwicki, and J. Schmidhuber, "Parallel multi-dimensional lstm, with application to fast biomedical volumetric image segmentation," *Advances in neural information processing systems*, vol. 28, 2015.
[58] D. Cheng and M. Liu, "Combining convolutional and recurrent neural networks for alzheimer's disease diagnosis using pet images," in *2017 IEEE International Conference on Imaging Systems and Techniques (IST)*. IEEE, 2017, pp. 1–5.
[59] R. P. Poudel, P. Lamata, and G. Montana, "Recurrent fully convolutional neural networks for multi-slice mri cardiac segmentation," in *Reconstruction, segmentation, and analysis of medical images*. Springer, 2016, pp. 83–94.
[60] D. J. Rezende, S. Mohamed, and D. Wierstra, "Proceedings of the 31st international conference on international conference on machine learning," 2014.
[61] J. Koutnik, K. Greff, F. Gomez, and J. Schmidhuber, "A clockwork rnn," in *International Conference on Machine Learning*. PMLR, 2014, pp. 1863–1871.
[62] Y. Xie, Z. Zhang, M. Sapkota, and L. Yang, "Spatial clockwork recurrent neural network for muscle perimysium segmentation," in *International Conference on Medical Image Computing and Computer-Assisted Intervention*. Springer, 2016, pp. 185–193.
[63] D. Ciresan, A. Giusti, L. Gambardella, and J. Schmidhuber, "Deep neural networks segment neuronal membranes in electron microscopy images," *Advances in neural information processing systems*, vol. 25, 2012.
[64] P. T. Truc, M. A. Khan, Y.-K. Lee, S. Lee, and T.-S. Kim, "Vessel enhancement filter using directional filter bank," *Computer Vision and Image Understanding*, vol. 113, no. 1, pp. 101–112, 2009.

Thuy Le-Nhi-Lam received B.S. and M.S. degree in Computer Science from Vienam National University – Hochiminh city, Vietnam in 2004 and 2009. She was lecturer in the Fundamental Science Department, HCMC College of Economics from 2005 – 2/2018. She have been lecturer in the Information Science Faculty, Sai Gon University, Vietnam. She is Ph.D. student of Industrial University of Hochiminh City, Vietnam. Her main research includes artificial intelligence, medical images, machine learning and data mining.
Email: thuylnl@sgu.edu.vn, ncs.lnlthuy@iuh.edu.vn

Sang Vu-Ngoc-Thanh received the B.Sc. degree in biomedical engineering from the International University–National University of HCM City, Vietnam, in 2014, the M.Sc. degree from the Department of Information Sciences, Tokyo University of Agriculture and Technology, Japan, in 2016, and the Ph.D. degree from the Department of Electronic and Information Sciences, Tokyo University of Agriculture and Technology, Japan, in 2019. Since 2019, he has been a Lecturer with the Computer Science Department, Saigon University, Vietnam. His main research includes artificial intelligence, medical images, machine learning and data mining
Email: vntsang@sgu.edu.vn

Hieu Huynh-Trung received the B.S. and M.S. in Computer Engineering from Ho Chi Minh city University of Technology in 1998 and 2003, respectively, and Ph.D. degree in Computer Engineering from Chonnam National University in 2009. He worked with the Faculty of Electrical and Electronics Engineering, Ho Chi Minh city University of Technology from 1998 to 2010. He is currently an associate professor in Industrial University of Ho Chi Minh city. His research interests focus on the computational intelligence for image analysis, pattern recognition, network and communication security, biological and medical data analysis. His main research includes medical image segmentation, artificial intelligence, machine learning and data mining
Email: hthieu@ieee.org

Bao Pham-The is an associate professor at Information Science Faculty, Sai Gon University. He received B.S., M.S., and Ph.D. degree in Mathematics and Computer Science from University of Science, Vietnam, in 1995, 2000, and 2009. He was lecturer in Department of Computer Science, Faculty of Mathematics and Computer Science, University of Science, Vietnam from 1995 to 2018. After 2018, he is lecturer in Information Science Faculty, Sai Gon University. His main research includes artificial intelligence, medical image processing, machine learning and data mining
Email: ptbao@sgu.edu.vn

INFORMATION FOR AUTHORS

Journal on Information Technologies & Communications or *ICT Research* in short – is a scientific publication of the Journal of Information and Communication, published by Vietnam Ministry of Information and Communications (MIC) since 1999. ICT Research is peer-reviewed, open-access, and published four times a year: two issues in *English* (ISSN: 1859-3534), and two issues in *Vietnamese* (ISSN: 1859-3526) (Chuyên san Các công trình nghiên cứu phát triển Công nghệ Thông tin và Truyền thông).

The purpose of *ICT Research* is to provide a forum for researchers and professionals to disseminate original and innovative ideas in the fields of information technology, communications, and electronics in Vietnam and worldwide. Its Editorial Board is composed of renowned scientists from research institutions throughout Vietnam and other countries.

AUTHOR GUIDELINES

Authors are encouraged to follow good practice of technical paper writing, such as the guidelines on "How to write for technical periodicals & conferences" of the Institute of Electrical and Electronics Engineers.

Submissions must be made online on the website of *ICT Research* [https://ictmag.vn]. Authors should format the manuscript as closely as possible to the *ICT Research* manuscript template.

By submitting a manuscript, authors are required to ensure that the submission has not been previously published, nor is currently submitted for publication elsewhere. Submission also implies that the corresponding author has the consent of all authors. The submission file must be in PDF format. To faciliate double-blind review, authors are requested to submit two versions of the manuscript, one with full information about all authors, the other without.

Submissions of revised manuscripts should follow the same guidelines as for the initial manuscript, plus with a document entitled "Response to reviewers' comments" which explains the modifications according to the comments of the anonymous reviewers.

Upon formal acceptance of a manuscript for publication, the authors are required to prepare the final manuscript in LaTeX, using the IEEE Transactions style.

PUBLICATION ETHICS

ICT Research is committed to following the rules, standards, and guidelines set forth by the Committee on Publication Ethics (COPE) in ensuring publication integrity and in handling cases of misconduct. All involved parties (Editors, Authors, Reviewers, and Publisher) are expected to be aware of and to conform to the respective rules, standards, and guidelines, available on COPE website.

The Editorial Board of *ICT Research* will immediately screen all articles submitted for publication. All submissions we receive are checked for plagiarism by using available online tools. Any type of plagiarism is unacceptable and is considered unethical publishing behavior. Such manuscripts will be rejected.

PEER REVIEW PROCESS AND PUBLICATION

An area editor of *ICT Research* (to be considered as the Editor from the point of view of the Authors) is assigned, by the Technical Editor-in-Chief, to each submission to coordinate the review process. The Editor pre-screens the submission and determines whether it is appropriate to the interests of readers of *ICT Research*. If appropriate, the Editor then initiates the review process. Each submission is then reviewed by at least two Reviewers. The review process is double blind, that is, identities of authors and reviewers are not revealed to each other.

Review decisions are: *(A) Accept submission* (i.e., no changes required), *(B1) Revision required* (i.e., minor revision), *(B2) Resubmit for review* (i.e., major revision, the submission needs to be re-worked with major changes and is then subject to a second round of review), and *(C) Decline submission* (i.e., reject). Each major round of review may take approximately 1 month. Should authors be requested by the Editor to revise the manuscript, the revised version should be submitted within 4 weeks for a major revision or 2 weeks for a minor revision. No more than 1 major revision and 2 minor revisions are allowed.

ICT Research is published with two issues a year in its *English Edition* (ISSN: 1859-3534). An accepted article will be immediately published online, with a DOI number, having been copyedited and laid out in compliance with *ICT Research* editing rules.

COPYRIGHT NOTICE