TELKOMNIKA Telecommunication, Computing, Electronics and Control
Vol. 18, No. 2, April 2020, pp. 830~838
ISSN: 1693-6930, accredited First Grade by Kemenristekdikti, Decree No: 21/E/KPT/2018
DOI: 10.12928/TELKOMNIKA.v18i2.14838

Single object detection to support requirements modeling using faster R-CNN

Nathanael Gilbert, Andre Rusli
Department of Informatics, Universitas Multimedia Nusantara, Indonesia

Article Info

Article history:

Received Jul 5, 2019
Revised Jan 7, 2020
Accepted Feb 19, 2020

Keywords:

Faster R-CNN
iStar 2.0
Object detection and recognition
Requirements modeling tool

ABSTRACT

Requirements engineering (RE) is one of the most important phases of a software engineering project in which the foundation of a software product is laid, objectives and assumptions, functional and non-functional needs are analyzed and consolidated. Many modeling notations and tools are developed to model the information gathered in the RE process, one popular framework is the iStar 2.0. Despite the frameworks and notations that are introduced, many engineers still find that drawing the diagrams is easier done manually by hand. Problem arises when the corresponding diagram needs to be updated as requirements evolve. This research aims to kickstart the development of a modeling tool using Faster Region-based Convolutional Neural Network for single object detection and recognition of hand-drawn iStar 2.0 objects, Gleam grayscale, and Salt and Pepper noise to digitalize hand-drawn diagrams. The single object detection and recognition tool is evaluated and displays promising results of an overall accuracy and precision of 95%, 100% for recall, and 97.2% for the F-1 score.

Corresponding Author:

Andre Rusli,
Department of Informatics,
Universitas Multimedia Nusantara,
Kampus UMN, Scientia Garden, Jl. Boulevard Gading Serpong, Tangerang, Banten, 15810, Indonesia.
Email: andre.rusli@umn.ac.id

1. INTRODUCTION

Broadly speaking, software systems requirements engineering (RE) is the process of discovering that purpose, by identifying stakeholders and their needs and documenting these in a form that is amenable to analysis, communication, and subsequent implementation [1]. The importance of RE is emphasized to develop effective software and reduce software mistakes in the early stage of software development [2]. Requirements modeling uses a combination of text and diagrammatic forms to depict requirements in a way that is relatively easy to understand, and more important, straightforward to review for correctness, completeness, and consistency [3]. In analyzing software requirements, after the domain is understood and elicited, requirements are evaluated and negotiated, then the consolidated requirements are specification specified and documented [4]. This requirements specification and documentation is where requirements modeling commonly occurs. Throughout requirements modeling, the primary focus is on what, not how, on iStar 2.0's strategic dependency model, the focus is on describing the dependency relationship between each actor in the system, along with the intentional elements. In the requirements engineering community, iStar 2.0 is gaining traction both in the academical and industrial fields and is used by many players in the community [5]. The framework is applied and implemented in various sectors, such as healthcare, security analysis, and eCommerce [6].

When modeling requirements and designing software products, many engineers still resort to drawing the diagrams manually by hand instead of using software tools. One reason could be that hand-drawing the diagrams could lead to more focused work and less distraction [7]. However, in a sustainable project with continuous revisions caused by requirements evolution, it gradually became apparent that the digitalization of the hand-drawn diagram is essential in an ever-evolving requirements engineering activities. One of the first steps in diagram digitalization is object detection and recognition. Object detection and recognition aim to detect and recognize every object belonging to a known class in an image [8]. Several pieces of research have shown the ability of the advanced neural networks in image/object recognition [9, 10, 11]; henceforth, this research meant to utilize neural network architecture to implement machine learning techniques to detect and recognize objects in the requirements diagram. In the machine learning field, the Region-based Convolutional Neural Network (R-CNN) architecture is a popular method with promising performance. The rapid growth has proposed the currently known Faster R-CNN (from its predecessors, the R-CNN, and the Fast R-CNN) with better accuracy and processing [12]. Other research also displays the potential of Faster R-CNN to detect an object in an image with high accuracy with the correct dataset [13].

Furthermore, image pre-processing also holds a vital role in processing datasets in object detection [14]. One standard process is the color-to-grayscale technique. Grayscale images are images with only have a single value for its every pixel, resulting in a grey image, which tends to be black on pixels with weak intensity and white on pixels with high intensity [15]. This research uses Gleam as the greyscaling method, as it is argued that compared to other techniques, Gleam performs better [16]. Furthermore, to perform upsampling of the dataset towards a high-performing model, Salt and Pepper noise is utilized for its ability to replicate image data with differences by inserting wrong bit transmission and analog to digital conversion [17].

This paper reports the result of the early study which aims to implement and evaluate the performance of Faster R-CNN, Gleam, and Salt and Pepper technique for single object detection and recognition in a hand-drawn iStar 2.0 strategic dependency model for requirements modeling. The model's performance is measured by calculating the precision, accuracy, recall, and F-measure when classifying the notation of iStar 2.0 symbols.

2. RESEARCH METHOD

In conducting the research to implement and evaluate the performance of Faster R-CNN, Gleam, and Salt and Pepper technique to for single object detection and recognition in a hand-drawn iStar 2.0 strategic dependency model for requirements modeling, the research methodologies are as follows.

- Literature review and requirements analysis,
- Experiment and system design,
- System construction and coding,
- Testing and evaluation, and
- Research documentation.

Firstly, literature review and requirements analysis activities are conducted to define the problem, then propose a solution, in this case, deciding the most suitable methods and practices. Secondly, after works of literature are reviewed, and problems are defined the architecture and system design is done using flowcharts to design the flow of the steps conducted in the object detection and recognition program and UI mockups for testing purposes. Then the designed system is constructed, and testing is conducted to evaluate the performance of the machine learning model. Lastly, all the activities conducted in the research is documented.

2.1. iStar 2.0

The i* language was presented in the mid-nineties [18] as a goal- and actor-oriented modeling and reasoning framework. It consists of a modeling language along with reasoning techniques for analyzing created models. i* was quickly adopted by the research community in fields such as requirements engineering and business mod- eling. Benefiting from its intentionally open nature, multiple extensions of the i* language have been proposed (see [19, 20] for useful reviews), either by slightly redefining some existing constructs, by detailing some semantic issues not completely defined in the seminal proposal, or by proposing new constructs for specific domains. As a response to the need of balancing the framework's open nature and a possible solution to the aforementioned adoption problems, the i* research community started an initiative to identify a widely agreed upon set of core concepts in the i* language. The main goal is to keep open the ability to tailor the framework while agreeing on the fundamental constructs, thus began

the work to propose an update to the framework, and to clearly distinguish this core language from its predecessors, it is named iStar 2.0.

2.1.1. iStar 2.0 elements

Actors are central to the social modeling nature of the language [6]. Actors are active, autonomous entities that aim at achieving their goals by exercising their know-how, in collaboration with other actors. Whenever distinguishing the type of actor is not relevant, either because of the scenario-at-hand or the modeling stage, the notion of generic actor-without specialization-can be used in the model. Actors are represented graphically as circles.

Intentional elements are the things actors want. As such, they model different kinds of requirements and are central to the iStar 2.0 language. An intentional element appearing inside the boundary of an actor denotes something that is desired or wanted by that actor. The following elements are included in the language [6], with examples shown in Figure 1:

- Goal: a state of affairs that the actor wants to achieve and that has clear-cut criteria of achievement.
- Quality: an attribute for which an actor desires some level of achievement. Qualities can guide the search for ways of achieving goals, and also serve as criteria for evaluating alternative ways of achieving goals.
- Task: represents actions that an actor wants to be executed, usually with the purpose of achieving some goal.
- Resource: A physical or informational entity that the actor requires in order to perform a task.

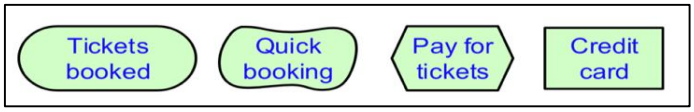


Figure 1. iStar 2.0 intentional elements [6]

2.2. Faster R-CNN, Gleam, and Salt and Pepper Noise

Faster Region-based Convolutional Neural Network is an upgraded version of R-CNN with a better performance for object detection. Figure 2 shows the architecture of Faster R-CNN, with steps as follows [21].

- Region Proposal Network: The very fast task is to search in the given input image the spaces where there is a probability of location of object.The position of the object in an image can be located [22]. These regions where there is possibility of object is bounded by a region known as region of interest(ROI).
- Classification: The stage is to classify the regions of interest identified in the above steps into corresponding classes.The technique deployed here is Convolution Neural Networks(CNN).
  In the proposed approach there is rigrous process of identifying all spaces of object location in image.However if no regions are identified in the first stage of algorithm then there is no need to further go to the second step of approach [23].

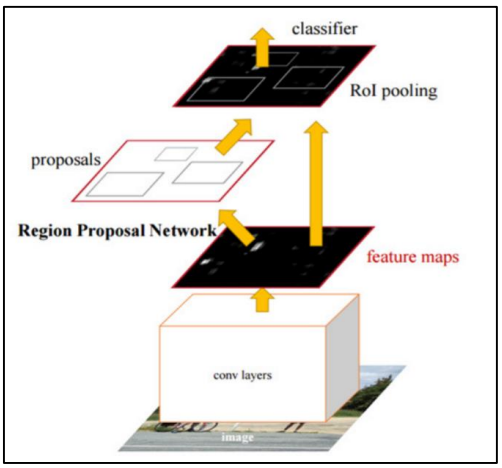


Figure 2. Faster R-CNN Architecture [21]

Color-to-grayscale is the transformation of RGB channel to an grayscaled image. Grayscale is the condition in which an image consist only a single value for each of its pixel. Grayscaled image generally consists of grey, black (in pixels with weak intensity), and white (in pixels with strong intensity) [15]. Formula (1) is the formula to convert the RGB channel in a pixel into a single value ranging from 0-255 (grayscale) [16], where the R', G', and B' are get from the RGB channels which are gamma corrected using Formula (2). Figure 3 shows the result of a grayscaling process using Gleam.

$$Gleam = \frac{1}{3}(R' + G' + B') \quad (1)$$

$$\Gamma(t) = t' = t^{1/2.2} \quad (2)$$

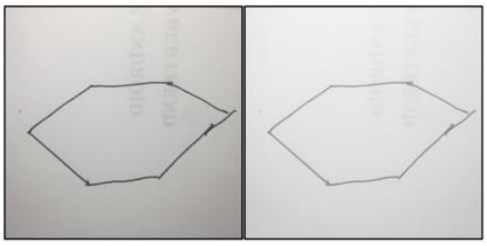


Figure 3. Example of grayscaling using Gleam

Salt and Pepper noise is used for replicating images in the dataset for training the model by applying noise in the original image. It does so by changing pixel value into the minimum or maximum value accepted [24, 25]. Figure 4 below shows the result of when we apply the noise into an image.

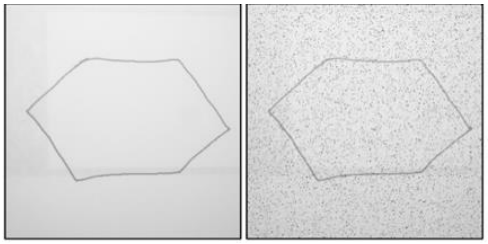


Figure 4. Application of Salt and Pepper Noise on a Hand-Drawn Task Object in iStar 2.0s

2.3. Requirements modeling tools

Several researches have already emphasized the importance i* framework [18] for modeling and documenting requirements, including the newly-standardized iStar 2.0 [6, 19, 20]. On previous researches, the proposal of integrating several requirements modeling framework and notation, including the early i* framework is conducted and showed the potential of using i* as a tool to model stakeholder dependency in analyzing early-phase requirements [26, 27]. Another research recognized the need of a tool for drawing and editing iStar 2.0 diagrams, then developed the piStar tool for supporting the creation of the requirements model [28]. Other researches proposed extensions to the iStar 2.0 [29] and prototype for generating meaningful layout [30]. However, the topic on digitalization and the use of machine learning architecture for object detection on iStar diagrams is still rare to be found. This research aims to address the missing topic by reviewing its importance and kickstarting the development of such tool.

2.4. Single object detection and recognition for iStar 2.0

Using the architecture provided by the Faster R-CNN technique, grayscaling using Gleam, and upsampling the dataset by replicating the image using Salt and Pepper, the program is then designed. Figure 5 shows the flow in which the training activity is done to built the machine learning model which will be used to detect and recognize objects. At the beginning, 600 image data are collected as the dataset, consisting of the drawings of 5 objects in the iStar 2.0 notation, goal, quality, actor, task, and resource. After the dataset is collected, labelling is done for each object, resulting in an XML file containing all the images and their labels. The generated XML is then converted to a CSV fille which then is used to train the machine learning model by running the export inference graph. These actions are done by utilizing

the TensorFlow Object Detection API. The resulting model is then tested and measured to find out the performance, based on a confusion matrix to calculate accuracy, recall, precision and the F-1 score of the model. Several configurations are tested to find the best scenario. The results are described in the next section of this paper.

Moreover, besides designing the training activity along with all its processes, the flow of the program feature which will detect and recognize submitted and unlabeled image data is also designed, as shown in Figure 6. The flow starts by getting the submitted image file of a hand-drawn iStar 2.0 object, then its pixels are converted and grayscaled (using Gleam). Furthermore, path initialization is done so that the developed object detection API knows the exact path of the file. NUM_CLASSES describes the number of existing classes in which an object will be classified to. An object is considered belonging to a class when it achieves a score bigger than 0.9, if there are more than one class that achieves 0.9, then the first identified class is considered as the correct class.

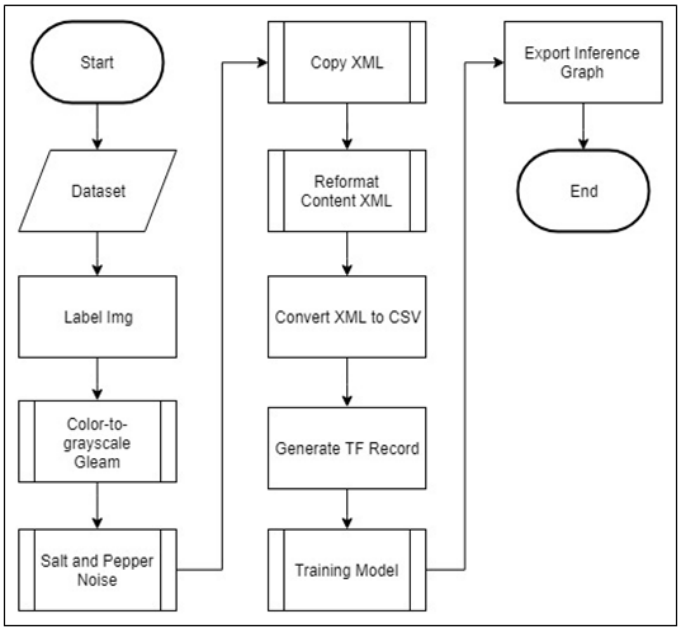


Figure 5. Flowchart training model

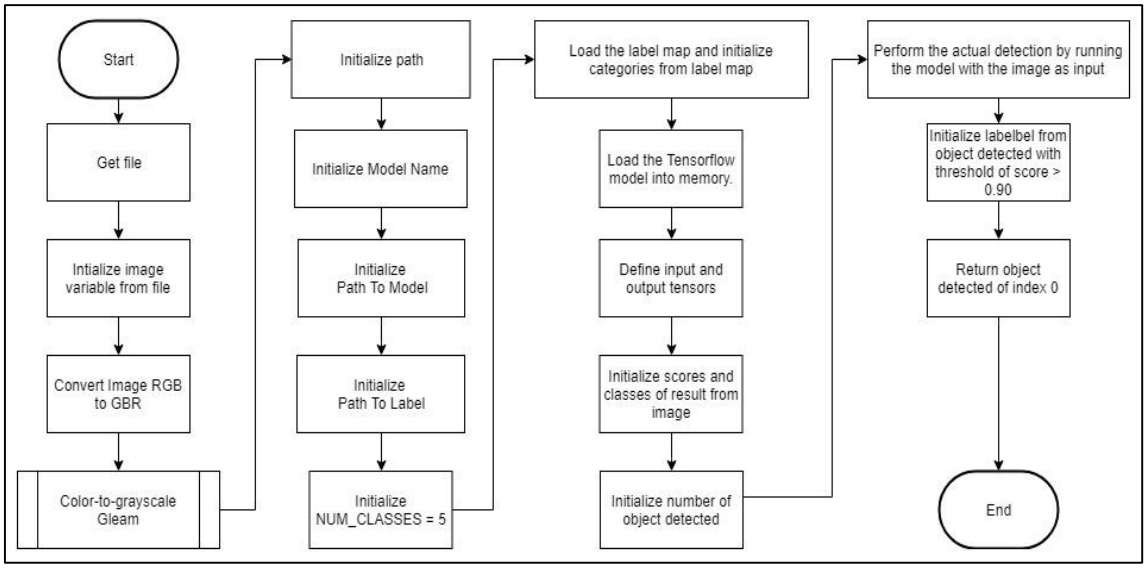


Figure 6. Flowchart iStar 2.0 object detection and recognition

3. RESULTS AND ANALYSIS

In order to measure the model's performance and evaluate its potential to be further developed as a support tool for requirements modeling, seven test scenarios were designed and experiments are conducted to find the best condition from all scenarios to build a high-performing model.

3.1. Test Scenarios

Various test scenarios are prepared by using various learning rate, feature extractor, initial crop size, maxpool kernel, and maxpool stride. The details of the seven test scenarios experimented can be seen in Table 1. From those seven scenarios, the model's ability to detect and recognize objects is then measured based on the confusion matrix result, calculating their accuracy, precision, recall, and F-1 Score.

Table 1. Test scenarios

Scenario	Learning Rate	Feature Extractor					Anchor Generator	
		Type	First stage features stride	Initial crop size	Maxpool Kernel	Maxpool Stride	Height Stride	Width Stride
1	0.0003	faster_rcnn_inception_resnet_v2	8	17	1	1	8	8
2	0.0002	faster_rcnn_inception _v2	16	14	2	2	16	16
3	0.0002	faster_rcnn_inception_resnet_v2	16	14	2	2	16	16
4	0.0001	faster_rcnn_inception_v2	16	14	2	2	16	16
5	0.0003	faster_rcnn_inception_v2	8	17	1	1	8	8
6	0.0003	faster_rcnn_inception_resnet_v2	8	14	2	2	8	8
7	0.0002	faster_rcnn_inception _v2	16	17	1	1	16	16

3.2. Result

After experiments are conducted based on the various configuration described in the previous section, test results are as described in Table 2. The highest performing scenario is found on the fourth scenario, using learning rate 0.0001, feature extractor type faster_rcnn_inception_v2 with 16 first stage features stride, 14 initial crop size, 2 maxpool kernel and stride, and 16 height and width stride, resulting in an average of 94% accuracy, 95% precision, 100% recall, and 97.2% F1-Score for each class.

Table 2. Test results

Scenario	Average			
	Accuracy	Precision	Recall	F1-score
1	57%	63%	97%	75,05%
2	94%	94%	100%	96,87%
3	42%	52%	95%	65,51%
4	95%	95%	100%	97,20%
5	94%	95%	97%	95,91%
6	39%	48%	96%	60,92%
7	88%	91%	99%	94,36%

From the results, it can be seen that the role of feature extractor, especially if we examine Scenario 1, 2, and 3, where feature extractor of type faster_rcnn_extractor_v2 performs much better than the other. Furthermore, initial crop size also proves to be quite impactful looking at Scenario 2 and 7. The learning rate can also be seen to be a determining factor even though it might not result in a big gap, between Scenario 4 and 2, for example.

In addition to the machine learning model as described above, our research also developed a simple web-based tool as the interface for users to demonstrate the model's ability to detect and recognize uploaded hand-drawn iStar 2.0 notions. Figure 7 shows the sample image that will be detected and recognized.

The notion depicted in Figure 7 is the Task in the iStar 2.0 strategic dependency model. Figure 8 displays the home page in which the Choose File feature can be clicked to upload the sample image, then when the Check Result button is clicked, the software will pre-process the image, detect the object, and determine if it is Task, Resource, Quality, Goal, or Actors, along with its match rate. Figure 9 shows the result, where the software is seen to be able to guess correctly what notion the uploaded image depicted, Task.

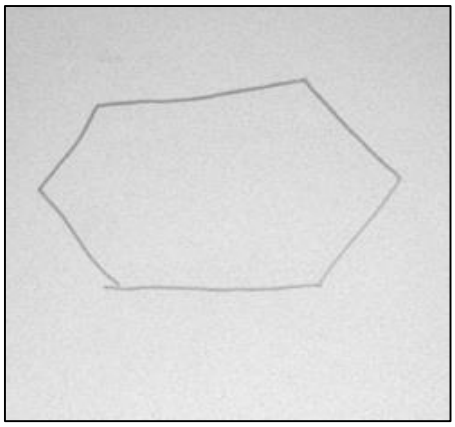


Figure 7. Sample Image for Object Detection

Figure 8. User interface of the web-based testing application

Figure 9. User interface when the application displays the object detection result

4. CONCLUSION

This research utilized Faster R-CNN using a dataset comprising of hand-drawn iStar 2.0 objects such as Generic Actor, Task, Resource, Quality, and Goal. Images are first pre-processed and replicated using Gleam for its color-to-grayscale technique and Salt and Pepper noise to give noise to the original dataset and duplicate the number of images in the dataset. The resulting program is best performing using 0.0001 learning rate, feature extractor type faster_rcnn_inception_v2 with 16 first stage features stride, 14 initial crop size, 2 maxpool kernel and stride, and 16 height and width stride, resulting in an average of 94% accuracy, 95% precision, 100% recall, and 97.2% F1-Score for each class. The conducted research displays the potential of Faster-RCNN, Gleam, and Salt and Pepper to build a model for detecting and recognizing objects drawn using the iStar 2.0 to enable the digitalization requirements diagram to support the requirements modeling activity in software development. Future works include improving the dataset and machine learning model to be able to digitalize a whole iStar 2.0 diagram, enabling the multi-object detection, and developing tools for editing and creating the whole diagram using the iStar 2.0 and other notation for requirements modeling. Optical character recognition techniques can also be integrated to be able to read texts inside the drawn objects.

ACKNOWLEDGEMENTS

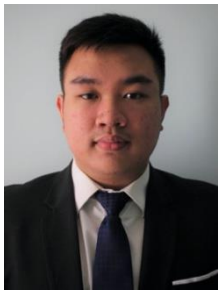
This research was supported by the Mobile Development Laboratory in Universitas Multimedia Nusantara. We also thank our colleagues from the Faculty of Engineering and Informatics who provided insight and expertise that greatly assisted the research, although they may not agree with all of the interpretations/conclusions of this paper.

REFERENCES

[1] Nuseibeh, B., Easterbrook, S., "Requirements Engineering: A Roadmap," *Proceedings of the Conference on the Future of Software Engineering. Limerick*, pp. 35-46, 2000.
[2] Chakraborty, A., Baowaly, M.K., Arefin, A., Bahar, A.N., "The Role of Requirements Engineering in Software Development Life Cycle," *Journal of Emerging Trends in Computing and information Sciences*, vol 3, no. 5, pp. 723-729. 2012
[3] Pressman, R.S., Maxim, B., "Software Engineering: A Practitioner's Approach," 8th Edition. New York: McGraw-Hill Education. 2014.
[4] Van Lamsweerde, A., "Requirements Engineering: From System Goals to UML Models to Software Specification," John Wiley & Sons Ltd: Chichester. 2009.
[5] X. Franch, A. Maté, J. C. Trujillo and C. Cares, "On the joint use of i∗ with other modelling frameworks: A vision paper," *2011 IEEE 19th International Requirements Engineering Conference*, pp. 133-142, Trento, 2011.
[6] Dalpiaz, F., Franch, X., Horkoff, J., "iStar 2.0 Language Guide,", arXiv preprint arXiv:1605.07767v3, 2016.
[7] Meltzer, L., "Executive Function in Education: From Theory to Practice," New York: The Guilford Press, 2007.
[8] Amit Y., Felzenszwalb P., "Object Detection". In: Ikeuchi K. (eds) Computer Vision. Springer, Boston, MA, 2014.
[9] Rahmat, R.F., Dennis, Sitompul, O.S., Purnamawati, S., Budiarto, R. "Advertisement billboard detection and geotagging system with inductive transfer learning" *TELKOMNIKA Telecommun Comput El Control*, vol. 17, no. 5, pp.2659-2666, 2019.
[10] Sudiatmika, I.B.K, Rahman, F., Trisno, Suyoto, "Image forgery detection using error level analysis and deep learning," *TELKOMNIKA*, vol. 17, no. 2, pp.653-659, 2019.
[11] Sugiarti, Yuhandri, Na'am, J., Indra, D., Santony, J., "An artificial neural network approach for detecting skin cancer," *TELKOMNIKA*, vol. 17, no. 2, pp.788-793, 2019.
[12] Jiang, H., Learned-Miller, E., "Face Detection with the Faster R-CNN," *12th IEEE International Conference on Automatic Face & Gesture Recognition (FG 2017)*, pp. 650-657, Washington, 2017.
[13] Kafedziski, V., Pecov, S., Tanevski, D., "Detection and Classification of Land Mines from Ground Penetrating Radar Data Using Faster R-CNN," *26th Telecommunications Forum (TELFOR)*, Belgrade. 2018.
[14] Pal, K.K., Sudeep, S., "Preprocessing for image classification by convolutional neural networks," *IEEE International Conference on Recent Trends in Electronics, Information & Communication Technology (RTEICT)*, pp. 1778-1781, Bangalore. 2016.
[15] Fatta, H.A. "Konversi Format Citra Rgb Ke Format Grayscale Menggunakan Visual Basic," *Seminar Nasional Teknolog*, Yogyakarta 2007.
[16] Kanan C, Cottrell GW., "Color-to-Grayscale: Does the Method Matter in Image Recognition?," PLoS ONE, 2012.
[17] Nazaré T.S., da Costa G.B.P., Contato W.A., Ponti M., "Deep Convolutional Neural Networks and Noisy Images," In: Mendoza M., Velastín S. (eds), "Progress in Pattern Recognition, Image Analysis," Computer Vision, and Applications. CIARP 2017. Lecture Notes in Computer Science, vol 10657. Springer, Cham. 2018.
[18] Yu, E.S.K., "Modelling strategic relationships for process reengineering," PhD *thesis*, University of Toronto. 1996.
[19] J. Horkoff et al., "Taking goal models downstream: A systematic roadmap," *IEEE Eighth International Conference on Research Challenges in Information Science (RCIS)*, pp. 1-12, Marrakech, 2014.

[20] Horkoff, J., Yu, E., "Comparison and evaluation of goal-oriented satisfaction analysis techniques," *Requirements Engineering*, vol. 18, no. 3, pp. 199-222, 2013.
[21] Abbas, S.M., Singh, S.N., "Region-based Object Detection and Classification using Faster R-CNN," *4th International Conference on Computational Intelligence & Communication Technology (CICT)*, Ghaziabad, 2018.
[22] Ren, S., He, K., Girshick, R., Zhang, X., and Sun, J., "Object detection networks on convolutional feature maps," *Coronell Universty, arXiv:1504.06066*, 2016.
[23] Szegedy, C., A. Toshev, and D. Erhan, "Deep neural networks for object detection," in *Neural Information Processing Systems (NIPS)*, 2013.
[24] Esakkirajan, S., Veerakumar, T., Subramanyam, A. N., Premchand, C. H., "Removal of High-Density Salt and Pepper Noise Through Modified Decision Based Unsymmetric Trimmed Median Filter," *IEEE Signal Processing Letters*, vol. 18, no. 5, pp. 287-290, 2011.
[25] Chan, R.H., Ho, C., Nikolova, M., "Salt-and-Pepper Noise Removal by Median-Type Noise Detectors and Detail-Preserving Regularization," *IEEE Transactions on Image Processing*, vol. 14, no. 10, pp. 1479–1485, 2005.
[26] Rusli, A., Shigo, O., "An Integrated Tool to Support Early-Phase Requirements Analysis," 4th International Conference on New Media Studies (CONMEDIA 2017), Yogyakarta, 2017.
[27] Rusli, A. Shigo, O., "Integrated Framework for Software Requirements Analysis and Its Support Tool," In: Requirements Engineering Toward Sustainable World: Third Asia-Pacific Symposium, APRES 2016, Nagoya, 2016.
[28] Pimentel, J., Castro, J., "piStar Tool – A Pluggable Online Tool for Goal Modeling," IEEE 26th International Requirements Engineering Conference (RE), Banff, 2018.
[29] Goncalves, E., Araujo, J., Castro, J. Towards Extension Mechanisms in iStar 2.0. iSTAR@CAiSE. Tallinn. 2018
[30] Wang Y., Li T., Zhang H., Sun J., Ni Y., Geng C. A Prototype for Generating Meaningful Layout of iStar Models. In: Woo C., Lu J., Li Z., Ling T., Li G., Lee M. (eds) Advances in Conceptual Modeling. ER 2018. Lecture Notes in Computer Science, vol. 11158. Springer, Cham. 2018.

BIOGRAPHIES OF AUTHORS



Nathanael Gilbert graduated from the Department of Informatics in Universitas Multimedia Nusantara, Indonesia, in November 2019. He has a background in research in applied machine learning for image processing and holds keen interests in the area of Android-based mobile application development.

Andre Rusli received his Master's of Science degree in Information Environment from Tokyo Denki University, Japan, in 2017, focusing in the Software Requirements Engineering field. He is currently a lecturer and researcher in Universitas Multimedia Nusantara and also serving as the head coordinator of the Mobile Development Laboratory. His research interests include requirements engineering in software application development, natural language processing, and human computer interaction.