

# Android-Based Application for Real-Time Indonesian Sign Language Recognition Using Convolutional Neural Network

Raymond Sutjiadi <sup>1</sup>

<sup>1</sup>*Department of Informatics, Institut Informatika Indonesia Surabaya, Jl. Pattimura No. 3, Surabaya, Indonesia*

**Abstract** – Individuals with hearing or speech impairments face challenges in communicating with others, requiring special techniques to express their thoughts and feelings. Sign language is an alternative way to communicate using a specific pattern of hand gestures to deliver messages instead of verbal speaking (oral communication). Unfortunately, most people do not know how to use and read sign language. Because of its complexity and many types of sign language worldwide, only well-trained personnel could use it as a communication medium. This research provides a solution in the form of a machine-learning Android-based application designed to recognize sign language captured by a smartphone camera and translate it into Latin characters. The recognition accommodates Convolutional Neural Network (CNN), one of the popular deep learning algorithms. This application recognizes 26 characters of Indonesian Sign Language (Bahasa Isyarat Indonesia/BISINDO) alphabets using MobileNetV3 architecture. To build the data model, dataset images were collected from 5 different models demonstrating 26 BISINDO characters in various lighting, background, and hand gesture position. These dataset images were also generated using image augmentation process to achieve the randomness by adjusting the image rotation, noise, and brightness. Based on the testing result using 6,240 dataset images, the application has 75.38% accuracy in recognizing Indonesian Sign Language alphabets.

DOI: 10.18421/TEM123-35

<https://doi.org/10.18421/TEM123-35>


**Corresponding author:** Raymond Sutjiadi,  
*Department of Informatics, Institut Informatika Indonesia Surabaya, Jl. Pattimura No. 3, Surabaya, Indonesia*  
**Email:** [raymond@ikado.ac.id](mailto:raymond@ikado.ac.id)

*Received:* 03 May 2023.

*Revised:* 02 August 2023.

*Accepted:* 07 August 2023.

*Published:* 28 August 2023.

 © 2023 Raymond Sutjiadi; published by UIKTEN. This work is licensed under the Creative Commons Attribution-NonCommercial-NoDeriv 4.0 License.

The article is published with Open Access at <https://www.temjournal.com/>

**Keywords** – Machine learning, convolutional neural network, MobileNetV3, sign language, BISINDO.

## 1. Introduction

Some people were born with physical impairment, posing challenges in their daily activities or interactions. Physical impairment could be classified into five categories, i.e., visual impairment, speech impairment, hearing impairment and equilibrium disturbance, visceral impairment, and mobility impairment [1]. More than 1,000 million people worldwide, or around 15% of the population, live with a disability [2]. Based on the Central Bureau of Statistics, Republic of Indonesia, in 2020, more than 22.5 million Indonesians, or 5% of the population, had physical impairment [3].

On the other hand, verbal communication involves the ability to speak and hear the conversation and to convey messages to each other. People express their ideas, thoughts, and feelings through verbal communication. Therefore, people with speech or hearing impairments are directly impacted by inability to communicate verbally. They are isolated and feel lonely because of the lack of communication. Therefore, this disability could lead to depression or mental health disorders [4] if there is no solution to bridge communication.

Sign language is a method to communicate with speaking/hearing-impaired people by expressing thoughts through manual (hand and body motion) or non-manual (facial expression) features [5]. Like spoken languages, many types of sign languages exist in society. Every country, region, or social community can adopt different sign languages [6]. For example, American Sign Language (ASL) is used in the United States (U.S.) and many parts of Canada; British Sign Language (BSL) is used in the United Kingdom (U.K.), and Chinese Sign Language (CSL or ZGS) is used in China.

In Indonesia, there are two commonly used sign languages, namely Indonesian Sign Systems (Sistem Isyarat Bahasa Indonesia/SIBI) and Indonesian Sign Language (Bahasa Isyarat Indonesia/BISINDO) [7].

SIBI is adopted from ASL, uses one-hand gestures, contains complex formal vocabularies, and has the prefix-suffix system. On the other hand, BISINDO is more familiar to be used among hearing-impaired communities in Indonesia because this sign language uses two-hand gestures, build naturally by the communities, and is more expressive in delivering messages [8].

Because of its complexity and variety, only a few people can use and communicate using sign language. Only well-trained experts can use it. This becomes an impediment to facilitating hearing-impaired communities to access public events and make interaction with society. Unluckily, Indonesia does not have adequate sign language training institutes. So far, only the Indonesian Sign Language Center (Pusat Bahasa Isyarat Indonesia/Pusbisindo) has conducted official sign language training.

Much recent research has been conducted to overcome this problem by building an intelligent application that utilizes machine learning to recognize sign language and convert it into Latin text. Kothadiya et al. [9] built a deep learning application called Deepsign to recognize Indian Sign Language, which utilized the combination of Long Short-term Memory (LSTM) and Gated Recurrent Unit (GRU). As the input, it used a video source, and the system generated the associated English word. To the experiment result, it achieved 97% accuracy over 11 different limited signs.

Another research conducted by Sundar and Bagyammal [10] . built an application to recognize American Sign Language using MediaPipe and LSTM. MediaPipe is an open-source library for capturing the hand key points. Meanwhile, LSTM was used to recognize the hand movement and classify it into a particular character. This research used sign language to represent 26 alphabet characters and showed 99% accuracy.

Gao et al. [11] researched to recognizing Chinese Sign Language by adopting the RNN-Transducer method.

H2SNet, an open Chinese dataset for continuous sign language recognition, was used as the training dataset.

This research outputted a lexical prediction from the sequential video input.

Indra et al. [12] proposed a method to detect hand shape features to recognize the BISINDO alphabet characters. This research used a webcam connected to a computer as an image acquisition process and had segmentation, edge detection, feature extraction, and, lastly, recognition process by calculating the difference in distance between the calculated feature with the shape feature in the database. The feature extraction process used the chain code based on the Freeman chain code model [13]. The experiment result showed 95% accuracy in recognizing 26 alphabet characters.

This research proposes a deep learning method using Convolutional Neural Network (CNN) to recognize 26 alphabet characters BISINDO based on an Android mobile application. The smartphone camera is used for image acquisition, which captures the hand gesture of BISINDO, then recognizes and translates it into 26 particular Latin alphabet characters. MobileNetV3 is implemented as the CNN architecture design tuned to perform optimal results for image detection and semantic segmentation on mobile phone CPUs [14].

## 2. Research Methodology

This section describes the method used in this research to define the system architecture and design precisely.

### 2.1. System Architecture

To develop an Android-based application for Indonesian Sign Language recognition using CNN comprises two modules system as shown in Figure 1.

#### A. Model Deployment Module

This module aims to train a data model using a training dataset. Model deployment is developed using Python programming language, Jupyter Notebook as the Integrated Development Environment (IDE), and Tensorflow Lite as the machine learning library. This module consists of several stages, as listed below:

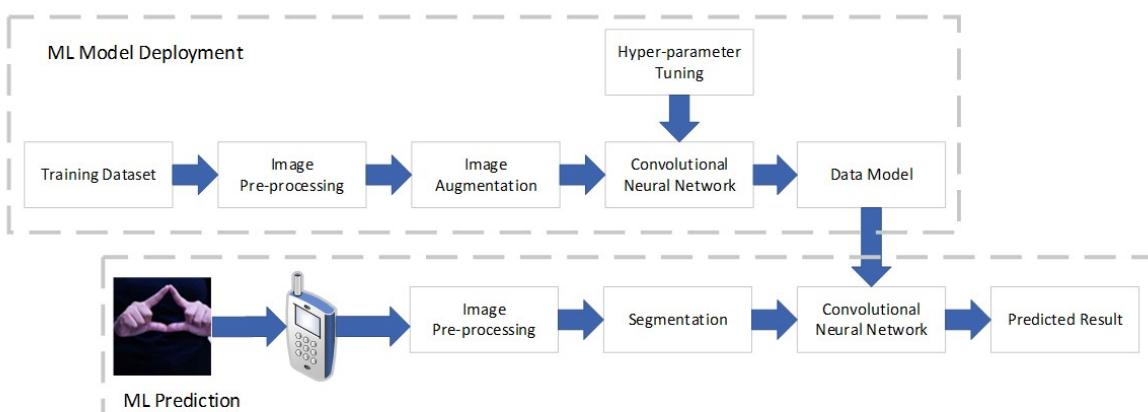


Figure 1. System architecture design

**i. Training Dataset**

This research collects the training dataset by capturing the sample images from 5 models demonstrating 26 BISINDO alphabet characters. To represent the variation of the dataset, five models wear different color shirts as the background, i.e., plain white, plain black, black with a motive image, and yellow with a motive image. The dataset is acquired by using Sony NEX-6 digital camera with a 35mm lens, f/5 aperture, and 1,080x1,920 pixel resolution. Image acquisition is held in a well-lit room using 12-watt (1100 lumens) LED lighting. A sample of the image dataset is shown in Figure 2.



Figure 2. Sample training dataset images

**ii. Image Pre-processing**

Raw training dataset images from the previous stage must follow the pre-processing stage. This image pre-processing is conducted by cropping and re-sizing the image resolution to 224x224 pixels, adjusted to the input image resolution of MobileNetV3 architecture. The cropping process is used to remove the unnecessary objects captured by the camera. In addition, the image brightness and contrast levels are adjusted to maintain image clarity before entering the following process.

**iii. Image Augmentation**

Image augmentation is a process to generate more training data from the original dataset images using image processing techniques [15]. This image augmentation stage is used to add variety to the training dataset, so it can improve the accuracy and minimize underfitting/overfitting conditions. In this research, image augmentation is performed by rotating the image up to 45 degrees clockwise and counterclockwise, adding random noise, and adjusting the image contrast on several levels.

The result of image augmentation is 240 training images per BISINDO alphabet character or 6,240 training images in total. This dataset will be divided into 70% for data training, 15% for data validation, and 15% for data testing.

**iv. Convolutional Neural Network**

Convolutional neural network is an artificial intelligence implementation with excellent performance in computer vision and machine learning problems [16]. In this research, MobileNetV3 is used as the CNN architecture, which is lightweight to be implemented on a mobile device yet achieves good accuracy compared to other more extensive networks [17]. Also, it is the latest improvement in architecture simplicity, speed (over 25% faster), and accuracy (6.6% more accurate) from the previous version (MobileNetV2) [14]. Figure 3 shows the general MobileNetV3 architecture block. In addition, Figure 4 shows that MobileNetV3's last stage layer is much simpler than the previous version.

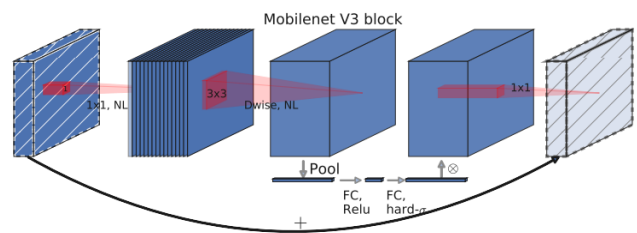


Figure 3. MobileNetV3 block [14]

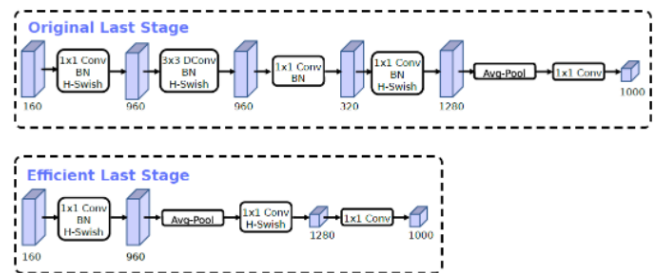


Figure 4. Last stage layer of MobileNetV2 comparing MobileNetV3 [14]

**v. Hyper-parameter Tuning**

This stage aims to optimize the accuracy by adjusting the parameters. Model development uses base learning rate = 0.001 with Adam optimizer. The batch size is 30, and the epoch is 500 iterations.

**vi. Data Model**

The output of this module is a classification data model. This model is used to find patterns of unseen data input based on the learning process of data training.

The best-expected training model is a model that has a high discriminative ability [18], which can minimize the underfitting and overfitting conditions.

## B. Prediction Module

This module aims to build an Android application to recognize and classify input data based on a data model produced by the model deployment module. This module uses Java programming language, Android Studio as Integrated Development Environment (IDE), Android Software Development Kit (SDK), and Tensorflow Lite as the machine learning library. This module consists of several stages, as listed below:

### i. Image Acquisition

During this stage, the user aims for a smartphone camera to capture the video of a person who demonstrates the hand gesture of a BISINDO alphabet character. The video must be taken in a well-lit condition to get a better result and focus the viewfinder on aiming the hand gesture while minimizing the other unnecessary objects. Choosing a background with plain color is recommended, e.g., wearing a shirt with simple color without any motives on it.

### ii. Image Pre-Processing

From the last stage, the video frame will be captured, and image pre-processing will be carried out. Only even frames will be taken to lighten the process and minimize oscillation on prediction results. Then, each frame is optimized by adjusting its contrast and brightness. This process ensures the image will fit the preliminary condition before entering the next stage.

### iii. Segmentation

Every image will enter the segmentation stage, where there is a process to detect a hand gesture, focus on it, and crop the image into 224x224 pixels resolution. This segmentation stage optimizes the input image by minimizing unnecessary objects interfering with recognition.

### iv. Convolutional Neural Network

This convolutional neural network stage will use the predicting data model already optimized and built by the deployment module as the pattern to detect and recognize a hand gesture. The Convolutional Neural Network uses the same architecture, MobileNetV3.

## v. Predicted Result

As a result, the recognition will be distributed into 26 classes (A until Z character). The biggest confidence score is taken as the valid predicted result. The label and confidence percentage will be shown on the application screen as the output result.

### 2.2. Use Case Diagram

Use case diagram identifies the primary function and services to be offered by the system [19]. Figure 5 below shows the use case diagram for this research. There are two actors, the user is the main actor who uses the application, and the target is the second actor who demonstrates sign language. The main actor will scan the sign language hand gesture demonstrated by the second actor using the camera of an Android device. The application then recognizes the hand gesture using the CNN model and outputs the result to the main actor as a predicted Latin alphabet and confidence score.

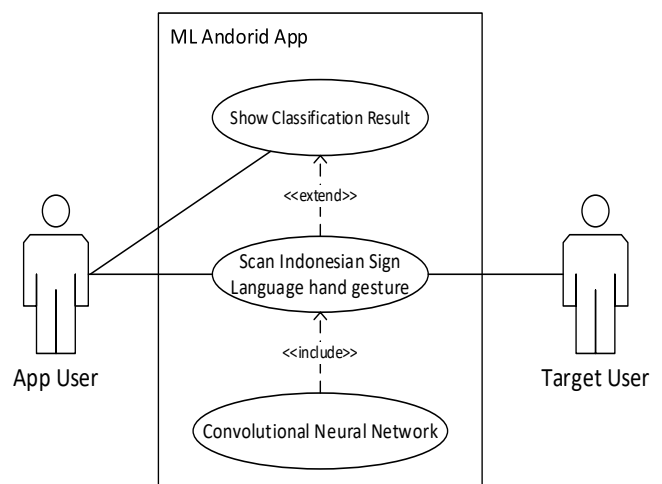


Figure 5. System use case diagram

### 2.3. Mobile Application User Interface

Figure 6 shows the design of the mobile application user interface. The main window shows the camera viewfinder aimed at hand gestures. At the top, the system prediction result will be displayed along with the confidence score. For example, Figure 6 shows the target user demonstrating BISINDO for character 'A' and the system gives a prediction result as character 'A' in the Latin alphabet with a 77% confidence score.

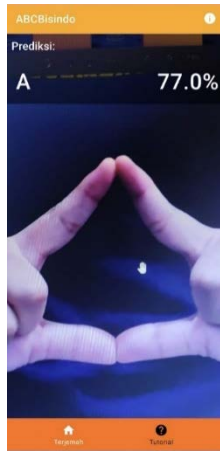











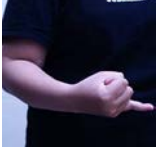

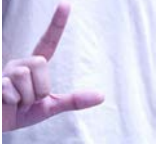
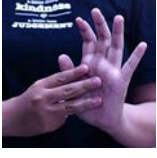

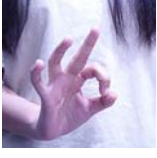

Figure 6. The design of mobile application interface



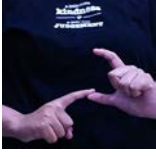
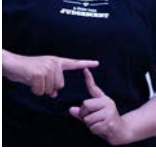
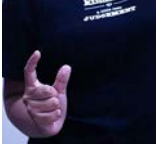


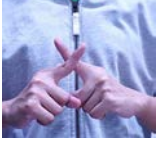


### 3. Results and Discussion

Five experiments are carried out to test the system's accuracy using five different models. Each model demonstrates 26 BISINDO alphabet characters from A to Z. The experiment result will be tabulated to determine whether the recognition succeeded or failed. The system accuracy is calculated from the total tests that correctly classify sign language (True Positive) divided by the total classification. The result of the per-character accuracy will be averaged to get the overall model accuracy. Table 1 shows the experiment results.

Table 1. Experiment results

No.	Testing Scenario	Expected Classification	Experiment Result					Accuracy Per Character
			T1	T2	T3	T4	T5	
1		A	A	A	A	A	A	100%
2		B	B	F	B	B	Z	60%
3		C	A	D	C	C	C	60%
4		D	A	D	D	D	A	60%
5		E	R	E	E	E	E	80%
6		F	E	F	F	F	F	80%

7		G	G	G	G	G	G	100%
8		H	E	H	H	H	H	80%
9		I	Z	I	I	I	I	80%
10		J	J	G	J	J	J	80%
11		K	A	K	D	K	K	60%
12		L	R	L	L	L	L	80%
13		M	M	M	M	M	M	100%
14		N	N	N	N	N	N	100%
15		O	R	C	O	O	O	60%
16		P	A	F	P	P	P	60%

17		Q	X	D	Q	Q	Q	60%
18		R	R	L	R	R	R	80%
19		S	S	G	S	S	S	80%
20		T	T	G	T	T	T	80%
21		U	R	L	U	U	U	60%
22		V	R	W	V	V	W	40%
23		W	R	W	W	W	W	80%
24		X	X	X	X	X	X	100%
25		Y	Y	F	W	Y	W	40%
26		Z	Z	Z	Z	Z	Z	100%
<b>Accuracy per Testing</b>			45%	53%	92%	100%	84%	75.38% (Average)

From the Table 1, it can be concluded that some characters have perfect accuracy, i.e., characters A, G, M, N, X, and Z. These characters have 100% accuracy from five testing results because BISINDO hand gestures for those characters are simple and quite different from other characters. On the other hand, some characters have relatively low accuracy, i.e., characters B, C, D, K, O, P, Q, U, V, and Y. These characters have 40%-60% accuracy from five testing results because BISINDO hand gestures for those characters are complex, some of them use two hands gesture, and have a similar form with other characters. The external conditions also influence the accuracy, for example, the lighting condition, the angle of the camera, and background complexity.

#### 4. Conclusion

Developing an Android-based application for real-time Indonesian Sign Language Recognition (BISINDO) can effectively help society interact with speaking/hearing-impaired people. This application aims to connect individuals in society who are unable to use or understand BISINDO with the ability to comprehend the intentions and messages of individuals who are speaking/hearing impaired and in such a way enhances the overall quality of life for those involved.

This research accommodates deep learning using Convolutional Neural Network (CNN) method with MobileNetV3 architecture to recognize 26 BISINDO alphabet characters and convert them into the Latin alphabet. From the experiment results it can be concluded that the average system accuracy from five experiment models reaches 75.38%. The use of MobileNetV3 on real-time Android application proves that this architecture is lightweight and adequate to maintain its accuracy without forfeiting reliability.

To enhance the accuracy value of this application, a recommendation for further improvement is to incorporate additional training dataset images that cover a wide range of conditions. Also, this application can be improved by adding the capability to recognize other BISINDO hand movements besides the alphabet characters. This application will significantly increase its functionality as a medium for recognizing BISINDO across the overall vocabulary.

#### References:

- [1]. Kohzuki, M. (2014). Classification of the Physical Disabilities and Actual Conditions of Visceral Impairment in Japan. *Asian Journal of Human Services*, 6, 125–137. Doi: 10.14391/ajhs.6.125
- [2]. World Health Organization. (2015). *WHO Global Disability Action Plan 2014-2021: Better Health for All People with Disability*. WHO Press.
- [3]. Purwaningsih, S. S., Budiarti, M., Yohanitas, W. A., Wulandari, P. R., & Citta A, G. (2022). *Naskah Kebijakan Pengembangan Riset Teknologi Alat Bantu bagi Penyandang Disabilitas: Rekomendasi Kebijakan Komite Nasional MOST-UNESCO Indonesia*. Penerbit BRIN. Doi: 10.55981/brin.679
- [4]. West, J. S. (2017). Hearing Impairment, Social Support, and Depressive Symptoms Among U.S. Adults: A Test of the Stress Process Paradigm. *Social Science & Medicine*, 192, 94–101. Doi: 10.1016/J.SOCSCIMED.2017.09.031.
- [5]. Papastratis, I., Chatzikonstantinou, C., Konstantinidis, D., Dimitropoulos, K., & Daras, P. (2021). Artificial Intelligence Technologies for Sign Language. *Sensors*, 21(17). Doi: 10.3390/s21175843.
- [6]. Lucas, C., & Bayley, R. (2011). Variation in Sign Languages: Recent Research on ASL and Beyond. *Linguistics and Language Compass*, 5(9), 677–690. Doi: 10.1111/j.1749-818X.2011.00304.x.
- [7]. Handhika, T., Zen, R. I. M., Murni, Lestari, D. P., & Sari, I. (2018). Gesture recognition for Indonesian Sign Language (BISINDO). *Journal of Physics: Conference Series*, 1028(1). Doi: 10.1088/1742-6596/1028/1/012173.
- [8]. Fadlilah, U., Mahamad, A. K., & Handaga, B. (2021). The Development of Android for Indonesian Sign Language Using Tensorflow Lite and CNN: An Initial Study. *Journal of Physics: Conference Series*, 1858(1). Doi: 10.1088/1742-6596/1858/1/012085.
- [9]. Kothadiya, D., Bhatt, C., Sapariya, K., Patel, K., Gil-González, A. B., & Corchado, J. M. (2022). DeepSign: Sign Language Detection and Recognition Using Deep Learning. *Electronics (Switzerland)*, 11(11). Doi: 10.3390/electronics11111780.
- [10]. Sundar, B., & Bagyammal, T. (2022). American Sign Language Recognition for Alphabets Using MediaPipe and LSTM. *Procedia Computer Science*, 215, 642–651. Doi: 10.1016/J.PROCS.2022.12.066.
- [11]. Gao, L., Li, H., Liu, Z., Liu, Z., Wan, L., & Feng, W. (2021). RNN-Transducer based Chinese Sign Language Recognition. *Neurocomputing*, 434, 45–54. Doi: 10.1016/J.NEUCOM.2020.12.006.
- [12]. Indra, D., Purnawansyah, Madenda, S., & Wibowo, E. P. (2019). Indonesian Sign Language Recognition Based on Shape of Hand Gesture. *Procedia Computer Science*, 161, 74–81. Doi: 10.1016/J.PROCS.2019.11.101.
- [13]. Jabr, Z. F. (2015). Hand Palm Recognition using Combination of Freeman Chain Code and Texture Features with Fuzzy Logic Classifier. *International Journal of Computer Science and Mobile Computing*, 4(3), 585–598.
- [14]. Howard, A. et al. (2019). Searching for MobileNetV3. *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, 1314–1324. Doi: 10.1109/ICCV.2019.00140.
- [15]. Shorten, C., & Khoshgoftaar, T. M. (2019). A Survey on Image Data Augmentation for Deep Learning. *Journal of Big Data*, 6(1). Doi: 10.1186/s40537-019-0197-0.
- [16]. Wu, J. (2017). Introduction to convolutional neural networks. *National Key Lab for Novel Software Technology. Nanjing University. China*, 5(23), 495.



- [17]. Qian, S., Ning, C., & Hu, Y. (2021). MobileNetV3 for Image Classification. *2021 IEEE 2nd International Conference on Big Data, Artificial Intelligence and Internet of Things Engineering (ICBAIE)*, 490–497. Doi: 10.1109/ICBAIE52039.2021.9389905.
- [18]. Dankers, F. J. W. M., Traverso, A., Wee, L., & van Kuijk, S. M. J. (2018). Prediction Modeling Methodology. In Kubben, P., Dumontier, M., Dekker, A. (eds) *Fundamentals of Clinical Data Science*, 101–120. Springer International Publishing. Doi: 10.1007/978-3-319-99713-1\_8.
- [19]. Aquino, E. R., De Saqui-Sannes, P., & Vingerhoeds, R. A. (2020). A Methodological Assistant for Use Case Diagrams. *Proceedings of the 8th International Conference on Model-Driven Engineering and Software Development, 1*, 227–236. Doi: 10.5220/0008938002270236.