Journal of information and communication convergence engineering
http://jicce.org

J. Inf. Commun. Converg. Eng. 18(3): 176-182, Sep. 2020 Regular paper

Implementation of an Open Artificial Intelligence Platform Based on Web and Tensorflow

Hyun-Jun Park and Kyounghee Lee*, *Member*, *KIICE*

Department of Computer Engineering, Pai Chai University, Daejeon 35345, Korea

Abstract

In this paper, we propose a web-based open artificial intelligence (AI) platform which provides high convenience in input data pre-processing, artificial neural network training, and the configuration of subsequent operations according to inference results. The proposed platform has the advantages of the GUI-based environment which can be easily utilized by a user without complex installation. It consists of a web server implemented with the JavaScript Node.js library and a client running the tensorflow.js library. Using the platform, many users can simultaneously create, modify and run their projects to apply AI functionality into various smart services through an open web interface. With our implementation, we show the operability of the proposed platform. By loading a web page from the server, the client can perform GUI-based operations and display the results performed by three modules: the Input Module, the Learning Module and the Output Module. We also implement two application systems using our platform, called smart cashier and smart door, which demonstrate the platform's practicality.

Index Terms: Artificial intelligence, Tensorflow.js, Node.js, Web-based platform

I. INTRODUCTION

With the recent emergence of popular artificial intelligence (AI) platforms such as Tensorflow [1-2] and Keras [3], AI services can be easily introduced into various research and business areas. Although such existing AI platforms provide potent means to benefit from AI technology, they are still not easy for inexperienced users to utilize. These platforms generally require complex installation procedures and sufficient expertise from the user to operate correctly. For example, users need to design and implement their own neural networks via programming languages such as Python and Java so that the machine can learn from the input data and generate the inference results. Moreover, these conventional platforms do not provide users with an easy means to deliver their results to control another system through open interfaces.

We therefore propose a platform in this paper that aims to retain the advantages of existing AI platforms while addressing the aforementioned drawbacks. The platform increases the convenience to users by providing easy installation, input data preprocessing, learning from the data, and configuring post actions based on the results. Because our platform is implemented on the web, it can provide a large number of users with open interfaces simultaneously within the same service environment.

The last of this paper is organized as follows. In Section 2, we introduce some existing well-known AI platforms and services with their distinguishing features. In Section 3, we propose our platform and explain its functional architecture in detail. Section 4 describes our implementation of the proposed platform. Two application systems implemented using our platform are also introduced. Finally, Section 5 concludes the paper.

Received 04 March 2020, **Revised** 11 September 2020, **Accepted** 15 September 2020
***Corresponding Author** Kyounghee Lee (E-mail: leekhe@pcu.ac.kr, Tel: +82-42-520-5341)
Department of Computer Engineering, Pai Chai University, Daejeon 35345, Korea.

Open Access https://doi.org/10.6109/jicce.2020.18.3.176 print ISSN: 2234-8255 online ISSN: 2234-8883

II. RELATED WORK

A number of well-known machine learning frameworks, such as Google's Tensorflow, Microsoft's CNTK [4], and PyTorch [5], have been implemented to facilitate the processes of acquiring data, learning models, and forecasting and refining future results. Users can design AI models and use learning functions based on hardware acceleration without understanding the core principles of machine learning or how the hardware works. These frameworks allow users to design and train neural network models for various purposes, however, they generally involve considerable programming work by the users at each step of the project to achieve their goals. These steps include input data pre-processing and visualization of the learned results, as well as the designing and training of the artificial neural networks.

IBM's Power Systems [6], Google's Google Cloud [7] and Amazon's AWS Cloud [8] allow users to operate machine learning frameworks with more abundant resources and in faster environments achieved by cloud computing. They generally provide higher computing performance for training neural networks compared to using a single computer such as a normal PC. Because these services are based on web platforms, they have the advantages of high accessibility and easy collaboration. These services are intended to help users implement their applications based on machine learning technology more conveniently; however, they still require much user programming work which many application software developers who are not AI experts are not familiar with.

The platforms described above are general-purpose AI platforms for developing various services. There are also many AI services with more specific services. Amazon's Alexa Voice [9] and Facebook's interactive AI, called the Messenger platform with Chatbot [10], can recognize speech or text and carry out user commands and pre-defined actions. IBM's Watson Health Platform [11] is a service that can be used for healthcare solution, including investigation of a user's health status using only a few body characteristics. These services are ready-to-use for users, but it is not possible for users to add or modify functions for their own additional purposes.

There are also some services that help users to easily add desired functions. Nugu Developus [12] is a web-based software development framework that enables a user to easily implement AI services running over Nugu, an AI speaker platform from SK Telecom. By simply combining functional blocks represented a GUI, the user can implement his own AI application program which performs suitable actions according to the situation. These applications can be shared through the user community. A user can search for and use applications by others if necessary. However, because Nugu Developus is not an open platform, it is not possible for the applications to be performed for various systems and devices. They also cannot interact with external systems easily.

Google's recent Teachable Machine [13] provides a web-based service which enables the user to easily experience machine learning applications. This service does not require the user to have any expertise in machine learning and has the advantage that it is very easy to use because of its intuitive user interface. It only provides three neural network models for classifying images, audio and human poses. But these models cannot be modified or upgraded by users. Moreover, the users are not allowed to build new neural network models.

Based on the above considerations of the existing approaches, we propose a web-based open platform that enables a user to easily implement artificial neural networks appropriate for various purposes. The key advantage of the proposed platform is that it provides a GUI-based development environment for the user to implement most application functions without programming. These functions include result visualization as well as basic neural network operations such as data input, output, and training. The user can implement additional functions such as input data preprocessing and pre-defining results-dependent operations if required, only by calling simple JavaScript API library functions provided with our platform. Users do not have to suffer the complex installation processes of existing general-purpose AI platforms and can easily utilize the desired functions with a web browser.

III. PROPOSED SYSTEM ARCHITECTURE

As shown in Fig. 1, the proposed platform consists of a web server implemented with the JavaScript Node.js library [14] and a client program running on the web browser. With the proposed platform, many users can simultaneously create, modify and run their projects to apply AI functionality to various smart services.

All operations, including artificial neural network design, learning from input data, and making inferences are performed through the web interface. Because our web server only provides the web interface to download the client program, most computations are performed on the user's computer, and the server is not involved in the client operations. Moreover, the system performance can be improved if advanced features in each web browser, such as GPU acceleration of Tensorflow.js [15], are enabled.

The main component of the web server is the Data Storage Module. The Data Storage Module stores user project data received from a client and provides the data when the client requests it later. Such user project data includes pre-processed (i.e., labeled) input data, artificial neural network models created by the user and JavaScript codes used for the

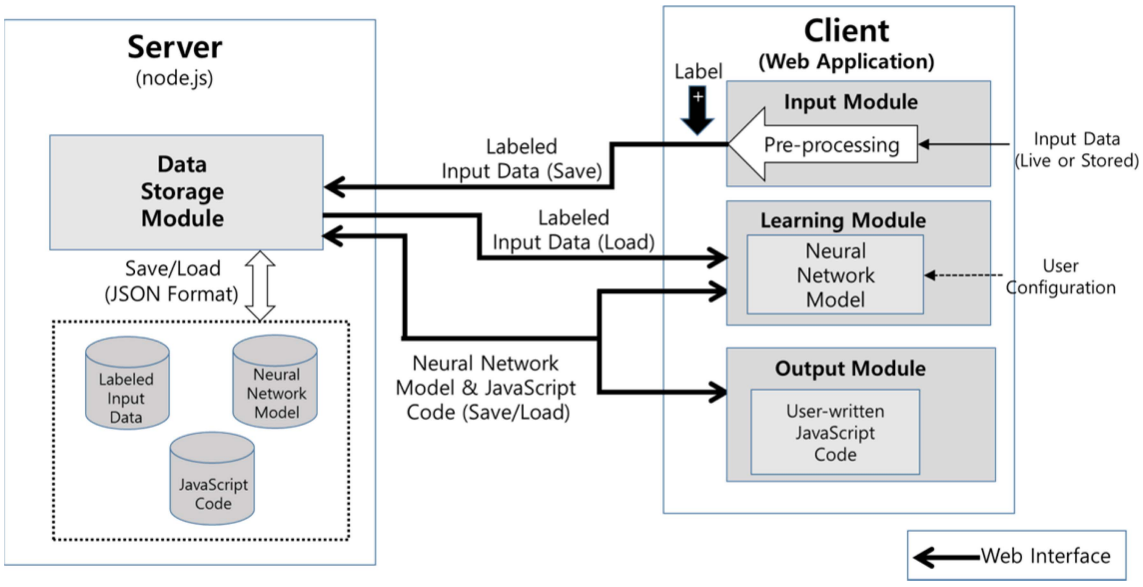


Fig. 1. Overview of the propose platform.

client application. The Data Storage Module stores the data in the JavaScript Object Notation (JSON) format [16].

The web client consists of three modules: the Input Module, the Learning Module, and the Output Module. The Input Module receives the user's input data and manipulates the data via pre-processing operations such as data labeling. Subsequently, the Input Module stores the labeled input data by sending it to the web server. The main role of the Learning Module is to train a neural network by making it learn from the labeled input data stored in the web server. Otherwise, the Learning Module makes an inference if the input data is given without labels. Another role of the Learning Module is to display the training status. The Output Module finally displays the learning results (i.e., inference) after the training is completed. Additionally, the Output Module enables a user to pre-define desired actions according to the inference results. After retrieving the inference from the artificial neural network, a pre-defined JavaScript code utilizing the inference results can be run. Accordingly, system's actions for specific results can be pre-configured so that depending on the results, the system performs various actions such as passing information to another system or directly controlling related hardware.

Fig. 2 shows the details of our client architecture. The Input Module can receive and manipulate input data provided by the user. The input data can be stored as files or take the form of live images generated by a camera device. The Input Module can perform pre-processing on the input data so that it is appropriate for training the Learning Module according to the pre-written JavaScript code. Finally, the Input Module transforms the input data into JSON format with data labeling before delivering it to the Data Storage Module in the server.

A user can choose a neural network model for the Learning Module by selecting one of the system's suggestions, which include well-known models such as the Convolutional Neural Network (CNN) and the Recurrent Neural Network (RNN). Otherwise, a user can design his own neural network with the help of various useful functions supported by the JavaScript Tensorflow.js library. After deciding the neural network model to be used, the Learning Module starts learning from the labeled input data stored in the Data Storage Module of the web server. As mentioned before, the Data Storage Module makes an inference if the input data is given without labels.

The inference results obtained from learning are delivered to the Output Module. The Output Module performs various tasks according to the results delivered by the Learning Module. These results can be directly presented in the web application's output window without any modifications. Otherwise, the results can be used as input to the pre-written JavaScript code so that the proposed platform performs desired actions depending on the results. The results can also be used to control IoT hardware with the JavaScript breakout.js library.

IV. IMPLEMENTATION RESULTS

Fig. 3 shows the user interface of the client program, which was implemented according to the functional architecture described in Section 3. As shown in Fig. 3, the client program mainly displays the operations and results of the three main functional modules: the Input Module, the Learn-

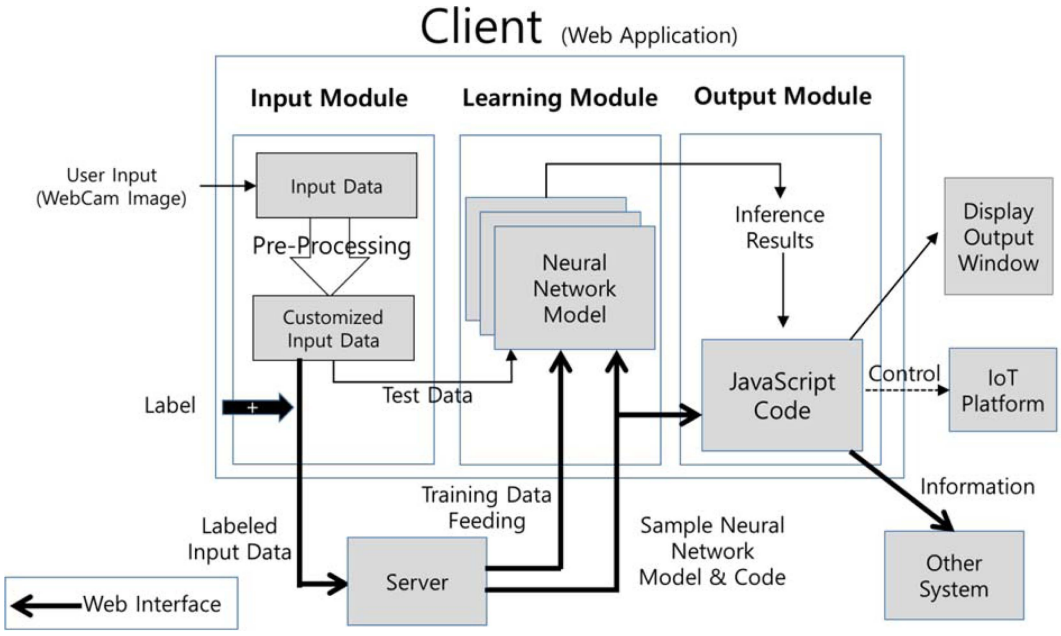


Fig. 2. Functional architecture of the proposed web client.

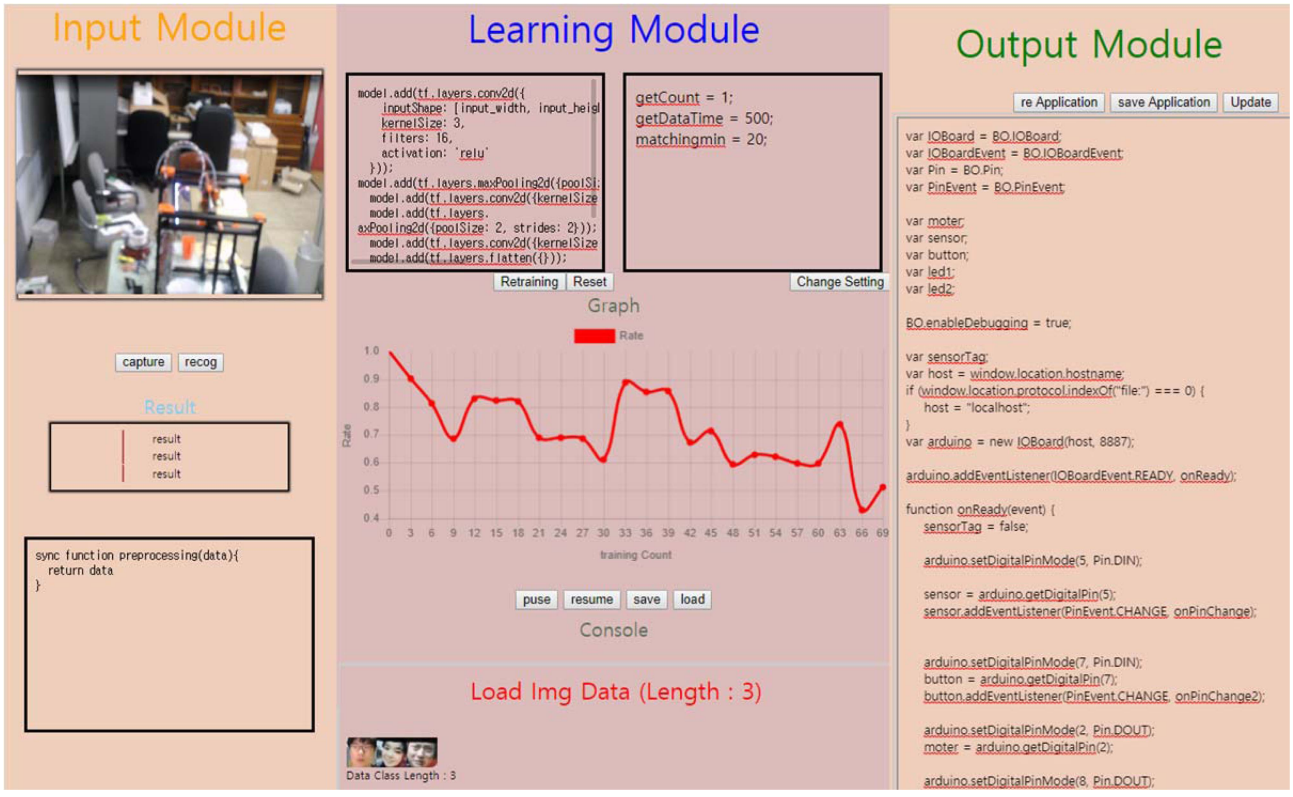


Fig. 3. User interface of the web client program.

ing Module, and the Output Module.

With the Input Module, a user can attach labels to the input data and periodically check the training status using the input data. The user can also write a JavaScript code to define operations for pre-processing the input data. The Learning Module allows users to design neural networks

with JavaScript code using the Tensorflow.js library and configure some parameters for neural network operations, such as the number of training stages (i.e., epoch and batch). The user is provided with some sample codes, such as a basic CNN model, which is useful for image training and classification. The Learning Module also displays a graph that shows the changes in the cost function while training a neural network model. In the Output Module, the user can see the results of the learning stage and write a JavaScript code to define post-actions according to the results. If necessary, the user can also configure the module to deliver these results to a predefined system.

We used the Node.js library to implement the web server of the proposed platform. Node.js is an event-handling I/O frame workload based on the V8 JavaScript engine, which helps users to create applications with JavaScript in a server environment. It exhibits high performance using a single-thread event loop. Tensorflow.js [15] has been used to perform AI operations in web applications. The machine learning models can be implemented and trained in a JavaScript environment. With Tensorflow.js, hardware acceleration using WebGL [17] can be applied and existing neural network models written in Python on the original Tensorflow platform can be used. Breakout.js [18] is a JavaScript library that enables a user to control IoT hardware directly. This is one of the functionalities implemented to allow web clients to perform various actions based on the neural network model's inference results. It uses the Firmata protocol [19] to access physical input and output through only JavaScript.

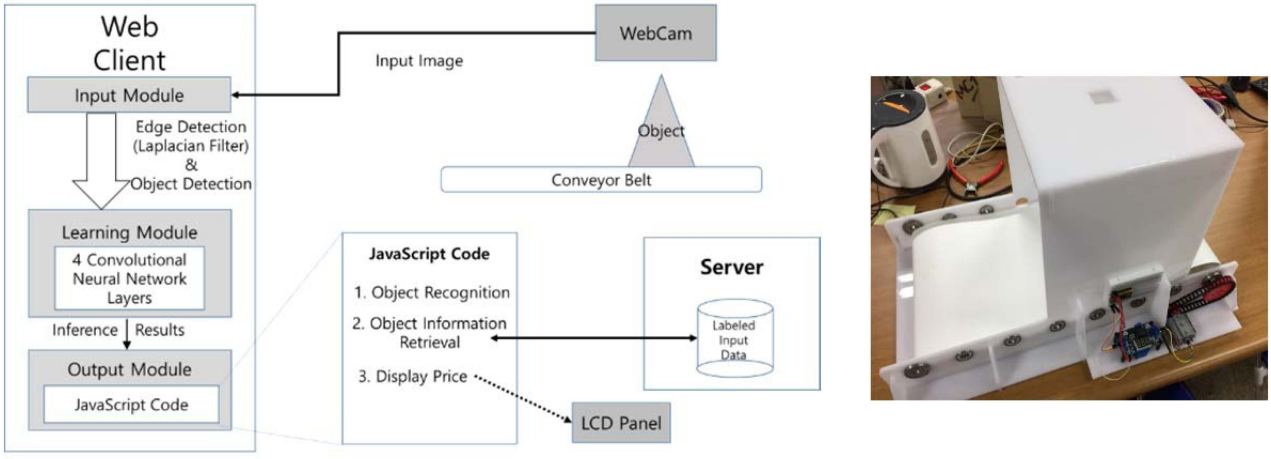
Communication between web servers and clients has been implemented with the Socket.io library [20]. Socket.io is a JavaScript library for web applications and enables real-time interactive communication between a server and clients. Socket.io mainly uses the WebSocket protocol [21]. It helps the user to easily utilize WebSocket and provides various functions such as broadcasting to multiple sockets and asynchronous I/O. The client has been implemented with the Embedded JavaScript Templates (EJS) [22] engine. The EJS code allows users to dynamically create web pages when creating HTML files. Because the web server performs rendering of the code and only the execution results of the code are displayed to users, the source code does not appear at the client side.

We used the d3.js module [23] to visually represent the cost function's loss value during the learning process. D3.js is a JavaScript library that helps the user to easily display graphs on a web page. The user can conveniently implement these graphs by simply entering matrix-type data.

Fig. 4 shows one of our application implementations, a smart casher system, which is based on the proposed AI platform. When a user places an object on the moving conveyor belt, the smart casher captures an image using the webcam and delivers the image to the Input Module of the web client. We use the Laplacian filter to detect the edges of the input image. The edges are used to determine the location of the target object. Accordingly, the detected object is categorized into one of the products by a 4-layers CNN, and finally, the Output Module runs the JavaScript code according to the inference result. Using the JavaScript code, the detailed data of the product is loaded from the server and displayed to show information such as the product's name and price.

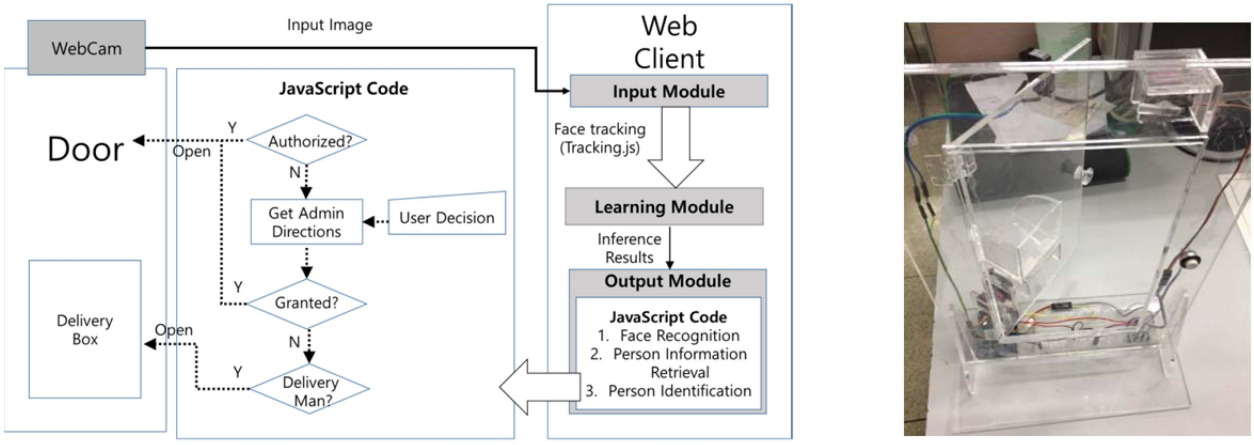
Fig. 5 shows another application system based on the proposed AI platform, which is a facial recognition smart door system. The system captures an image of a visitor at the door using a webcam and delivers the image to the web client's Input Module. The JavaScript library Tracking.js [24] is used to detect the area of the human face from the input image. Then, the detected face image is used by a 4-layers CNN to determine whether the visitor is an authorized person.

The inference result is given as the name of the visitor



(a) System overview (b) Implementation

Fig. 4. Applications I - Smart casher system based on object recognition.



(a) System overview (b) Implementation

Fig. 5. Applications II - Smart door system based on person identification.

(i.e., a label of the face image). By searching the label from the list of authorized persons, the client determines who the visitor is. If the visitor is an authorized person, the door will be opened. If the visitor is not authorized, the system immediately generates a message including the image of the visitor's face to inform the visit. Then, the system pushes the message to the administrator's computer or mobile device so that the administrator can directly check the image and decide whether to open the door (including the delivery box door).

V. CONCLUSION

In this paper, we proposed a web-based open AI platform that enables a user to easily implement artificial neural networks appropriate for his purpose. The proposed platform has the advantage that a user can easily utilize GUI-based functions without complex installation or much knowledge of the AI framework. This platform is ready for use by a user with simply running our program on the web server.

Our client program is a web application consisting of three functional modules, namely, the Input Module, the Learning Module, and the Output Module. With the client modules, users can conveniently design and train their own artificial neural networks. They can also define various reactive actions according to the inference results of the neural network by writing corresponding JavaScript codes. We also introduced two application systems implemented with the proposed AI platform: a smart cashier and a smart door system based on object/face recognition.

One of our future work would be to enhance the proposed platform so that it provides an API library implementing some frequently used pre-processing functions such as object and face recognition. Another enhancement would be to improve the user interface so that a user can utilize the proposed platform more easily.

ACKNOWLEDGEMENTS

This work was supported by the research grant of Pai Chai University in 2019.

REFERENCES

[1] M. Abadi, P. Barham, J. Chen, Z. Chen, A. Davis, J. Dean, M. Devin, S. Ghemawat, G. Irving, M. Isard, M. Kudlur, J. Levenberg, R. Monga, S. Moore, D. G. Murray, B. Steiner, P. Tucker, V. Vasudevan, P. Warden, M. Wicke, Y. Yu, and X. Zheng, "Tensorflow: a system for largescale machine learning," in *Proceeding of the 12th USENIX Symposium on Operating Systems Design and Implementation*, Savannah: GA, pp. 265-283, 2016. DOI: 10.5555/3026877.3026899
[2] Google, Tensorflow: An end-to-end open source machine learning platform [Internet], Available: https://www.tensorflow.org/.
[3] Keras [Internet], Available: https://keras.io/.
[4] Microsoft, The Microsoft Cognitive Toolkit [Internet], Available: https://docs.microsoft.com/en-us/cognitive-toolkit/.
[5] PyTorch [Internet], Available: https://pytorch.org/.
[6] IBM, IBM Power Systems [Internet], Available: https://www.ibm.com/kr-ko/it-infrastructure/power/.
[7] Google, Google Cloud [Internet], Available: https://cloud.google.com/.
[8] Amazon, AWS Cloud [Internet], Available: https://aws.amazon.com/.
[9] Amazon, Alexa Voice Service [Internet], Available: https://developer.amazon.com/en-US/alexa/alexa-voice-service/.
[10] Facebook, Messenger Platform with Chatbot [Internet], Available: https://developers.facebook.com/docs/messenger-platform/.
[11] IBM, Watson Health [Internet], Available: https://www.ibm.com/watson-health/.
[12] SK Telecom, Nugu Developus [Internet], Available: https://developers-doc.nugu.co.kr/.
[13] Google, Teachable Machine [Internet], Available: https://teachablemachine.

withgoogle.com/.
[14] Node.js Foundation, Node.js: JavaScript runtime built on Chrome's V8 JavaScript engine [Internet], Available: https://nodejs.org/.
[15] Google, Tensorflow.js [Internet], Available: https://www.tensorflow.org/js/.
[16] T. Bray, "JavaScript Object Notation," *Internet Standard*, IETF RFC 8259, 2017, [Online] Available: https://www.rfc-editor.org/pdfrfc/rfc8259.txt.pdf.
[17] Khronos Group, WebGL [Internet], Available: https://www.khronos.org/webgl/.
[18] Breakout Forum, Breakout.js [Internet], Available: http://breakoutjs.com/.
[19] GitHub, Firmata protocol [Internet], Available: https://github.com/firmata/protocol/.
[20] Socket.io Community, Socket.io 2.0 [Internet], Available: https://socket.io/.
[21] I. Fette and A. Melnikov, "WebSocket protocol," *Internet Standard*, IETF RFC 6455, 2011, [Online] Available: https://www.rfc-editor.org/pdfrfc/rfc6455.txt.pdf.
[22] Embedded JavaScript Templating (ETS) [Internet] Available: https://ejs.co/.
[23] GitHub, D3: Data-Driven Documents [Internet], Available: https://github.com/d3/d3/.
[24] Tracking.js [Internet], Available: https://trackingjs.com/.

Hyun-Jun Park
received a B.E. from Dept. of Computer Engineering in Pai Chai University, Daejeon, Republic of Korea in 2018. Now he is a M.E. student at the Dept. of Computer Engineering in Pai Chai University. His research interests include Web based AI platforms, generative adversarial neural networks, and reinforcement learning.

Kyounghee Lee
received his B.S. degree in Computer Science from Kwangwoon University, Seoul, Korea, in 1999. He received his M.S. and Ph.D. degrees from the Korea Advanced Institute of Science and Technology (KAIST), Daejeon, Korea, in 2000 and 2006, respectively. From 2006 to 2013, he was a senior researcher in the Electronics and Telecommunications Research Institute (ETRI), Daejeon, Korea. Since 2013, he has been a faculty member of Department of Computer Engineering in Pai Chai University, Daejeon, Korea. His research interests include future networks, mobile communications, Internet of Things, mobile software and real-time multimedia.