Sensors and Materials, Vol. 34, No. 3 (2022) 1229–1240
MYU Tokyo

S & M 2886

Flexible IoT Cloud Application for Ornamental Fish Recognition Using YOLOv3 Model

Chi-Tsai Yeh,[1] Tzuo-Ming Chen,[2*] and Zhong-Jie Liu[1]

[1]Changzhou College of Information Technology,
No. 22 Mingxin Middle Road, Science Education City, Changzhou City, Jiangsu Province 213164, China
[2]Shih Chien University, No. 200 University Road, Neimen, Kaohsiung City 84550, Taiwan, R.O.C.

(Received August 2, 2021; accepted February 14, 2022)

Keywords: microservices, deep learning, cloud computing, Internet of Things, YOLOv3

The ornamental fish industry is a booming and emerging industry. Fish identification for fish farmers, trainers, sellers, and even buyers is an essential skill. With the rise of deep learning, object recognition is widely used in different fields. In this paper, we propose a highly flexible application via microservice architecture. It utilizes cloud computing to provide high-efficiency and low-cost identification services with NVIDIA T4 graphics processor units (GPUs) and introduces the You Only Look Once (YOLOv3) model to recognize the species of ornamental fish. Finally, mobile devices capture the photos and communicate with cloud through a representational state transfer (RESTful) application programming interface (API) to retrieve the predicted results. The proposed application identified 11 types of ornamental fish and completed each prediction within 1 s.

1. Introduction

The size of the trade in global ornamental fish has steadily increased in recent decades. There were only 28 countries exporting ornamental fish in 1976, compared with more than 125 countries today. Moreover, the global export market for ornamental fish almost doubled from US$177.7 million in 2000 to US$337.7 million in 2016.[1,2]

The global ornamental fish industry is worth US$15 billion, and more than 2 billion live ornamental aquatic animals are traded per year, making it necessary to consider how to deal with large numbers of live ornamental aquatic animals. Until now, the calculation of the numbers of live fish and shrimps has mainly been carried out manually, which has many weaknesses, including subjectivity, slowness, low accuracy, worker fatigue, and even injury and death of animals during the calculation.[3]

In this study, we came up with a solution to recognize various fishes, which helps related users such as fish farmers, traders, and buyers identify fishes conveniently by integrating emerging technologies, namely, Internet of Things (IoT), cloud computing, and artificial intelligence, to complete this task. The interconnected IoT-cloud platform offers a highly flexible, on-demand, and cost-efficient hyperscale architecture. The developers can leverage the

*Corresponding author: e-mail: jchen@g2.usc.edu.tw
https://doi.org/10.18494/SAM3574

ISSN 0914-4935
https://myukk.org/

significant potential of IoT without having to build the underlying infrastructure and services. In this research, we adopted Amazon Web Service (AWS) for cloud computing and mobile phones as IoT devices. Mobile phones capture the images and receive the recognized results, and AWS services process the requests and recognize the images.

We faced some challenges concerning the integration of the proposed application. The first one is how to develop this application. Most companies have recently transferred their applications from a monolithic architecture to a cloud-based microservice architecture. The development of these applications with microservice architecture is more complicated and expensive in the beginning. The main advantages and disadvantages of the microservice architecture studied are listed below.[4–7]

Advantages:

- Loose coupling: the application minimizes the dependences between two or more components.
- Problem isolation: problems do not directly affect others.
- Service dependence: each service is a self-contained service, so the developing team is responsible for one or multiple services.
- Change-free: when a developer issues updates to a certain microservice module, the whole application does not need to be re-deployed.

Disadvantages:

- The structure of the application is more complex than that of monolithic application.
- However, these additional network partitions have been introduced in integration test, making the testing strategies are more difficult to perform than with monolithic applications.
- A larger development team is needed because each service can have its own technology.
- Initial application development is slow.

In accordance with the nature of this application, we adopted the cloud-based microservice architecture in this study. The remainder of this paper is organized as follows. In Sect. 2, we describe related work on machine learning and deep learning in aquaculture. In Sect. 3, we show the hardware and software elements of the proposed system. In Sect. 4, we explain the process flow used with our proposed method. In Sect. 5, we report experimental results obtained with the implemented system. Finally, we conclude the article in Sect. 6.

2. Related Work

2.1 Machine learning of computer vision in aquaculture

Zion summarized the applications of computer vision technology in aquaculture into five categories. His review described the state of the art and focused on needs in all stages of the aquaculture process from hatcheries to harvesting. These applications included counting, size measurement and mass estimation, gender detection and quality inspection, species and stock identification, and monitoring of welfare and behavior. Optical methods to count objects were the most common approach. The counted objects included eggs, larvae, fry, and fish at various stages of growth.[8] Silvério categorized these computer vision systems in aquaculture. These systems have the following common steps.[9]

- Capturing images
- Extracting features
- Detecting blobs
- Counting based on the blobs detected.

Most computer vision applications that perform image recognition require the above four steps. The image processing techniques are mostly used in steps 2 and 3, i.e., extracting features such as boundaries or a given color range and detecting blobs to segment the image. In this paper, we present a state-of-the-art application of computer vision for counting in aquaculture. A comprehensive summary of related studies for object identification or counting in aquaculture from 1995 to 2018 is given in Table 1.

Newbury *et al.* used artificial fish to simulate the process of fish calculation, especially considering the fact that fish often overlap with each other.[10] Then, they applied a backpropagation algorithm and achieved 94% accuracy when estimating the number of fish. Huang adopted the value of intensity (V) in the HSV color space to filter the target image, then performed thresholding for image segmentation, and calculated the area of fry raised under a fluorescent lamp on the basis of the obtained foreground.[11] The depth of each fry in an aquarium is different, resulting in fish of the same size occupying areas of different sizes in an image. In addition, the overlapping of fry was not considered, so errors of between 0.9 and 32% arose. In 2005, Friedland *et al.* used the mathematical morphological dilation–erosion algorithm to eliminate touching eggs, and then performed seven measurements to identify American shad eggs and debris in the water, enabling them to count the number of American shad eggs.[12] The false negative (FN) rate in the identification, i.e., eggs judged to be debris, was 1%. Zion developed image processing algorithms using shape models of various species and detected the number of overlapping fish fry with 98% accuracy.[8]

Table 1
Related literature on the application of computer vision to object identification or counting in aquaculture.

Authors	Applications	Targets	Results
Newbury *et al.*, 1995[10]	Backpropagation network	Artificial fish	94% accuracy, overlapping
Huang, 2002[11]	HSV background subtraction	Fluorescent-lamp fry	Error ranging between 0.9 and 32%, non-overlapping
Friedland *et al.*, 2005[12]	Erosion–dilation filter	American shad eggs	FN classification error of 1%, non-overlapping
Zion, 2012[8]	Shape model	Fish fry	98% accuracy, overlapping
Flores *et al.*, 2008[13]	Adaptive threshold and fast Fourier transform (FFT)	Peruvian scallop larvae	Accuracy of 95% in 2 min, overlapping
Khantuwan and Khiripet, 2012[14]	Adaptive threshold and co-occurrence color of histogram	Shrimp larvae	Accuracy of 97% in 1 min, overlapping
Coronel *et al.*, 2018[15]	Local normalization filter and iterative selection threshold	Fish fingerlings	Precision, recall, and F measure of 99.80, 97.90, and 98.83%, respectively, in <1 s, non-overlapping

2.2 Deep learning of computer vision in aquaculture

Deep learning is a type of machine learning and is considered an improvement of machine learning. According to the studies mentioned earlier, the researchers needed to find the features of the targets themselves. However, deep learning uses a programmable neural network that enables machines to find the features and then make accurate decisions without help from humans. Object detection algorithms can help users find the locations of the objects (stage one) and recognize the objects (stage two) in a photo. The convolution neural network (CNN) is the most popular method in object detection. There are two types of object detection algorithm: one-stage object detection, which finds the locations and recognizes the objects in one stage, represented by the YOLO series and single-shot detector (SSD), and two-stage methods represented by region-based convolutional neural networks such as R-CNN, Fast-RCNN, and Faster-RCNN. In 2016, Redmon *et al.* proposed the YOLO method to treat object detection, which is faster but less accurate than two-stage methods.[16] Redmon and Farhadi improved YOLO (v2) in handling images of different sizes, object positioning accuracy, and small object detection. They replaced the fully connected layer of the original YOLO with a fully convolutional layer and introduced anchor boxes and multiscale detection.[17] The authors proposed YOLOv3 in 2018 and considerably improved the detection of smaller objects in comparison with YOLOv2 in 2017.[18] Because of the excellent performance of YOLOv3, we applied it to our proposed application.

2.3 IoT in aquaculture

The applications of IoT in aquaculture include water quality monitoring/control,[19–23] automatic feeding,[19,23] disease detection,[21,22] and quantity calculation.[24] In 2017, researchers from National Chiao Tung University (NCTU) developed an IoT platform called IoTtalk. In 2019, Lin and Tseng introduced the FishTalk system based on IoTtalk, which utilizes aquarium sensors to monitor and improve the water conditions by driving actuators in real time.[19] They also implemented an intelligent fish feeding mechanism called FishFeeder for miniature aquarium tanks. To prevent forgotten feeding of fish, FishFeeder automatically feeds fish every 24 hours. Fish owners can trigger FishFeeder through a smartphone if the fish have not been fed for at least 12 hours. In 2019, Rohit *et al.* introduced a system that has a camera that takes underwater videos in real time.[21] The proposed system also has a single-board computer, NodeMCU, that processes images from the underwater camera. It provides functions such as fish tracking, fish counting, fish size measurement, and fish disease detection for mainly the ulcerative syndrome (EUS) symptom, which manifests as tiny red spots on the fish body surface. Rohit *et al.* used the HSV color histogram for disease detection. Agossou (2021) used the Alexnet convolutional neural network (CNN) to classify diseases such as Columnaris, Ichthyophthirius (ICH), and EUS.[22] Yeh and Ling applied the HSV color histogram to count the number of shrimps using Raspberry Pi.[24] Van *et al.* demonstrated the PlantTalk intelligence based on IoTtalk.[25] The users flexibly configure the connections of various plant sensors and actuators via a smartphone. Clearly, the application of IoT in aquaculture has gradually changed from the

early simple image processing to complex image recognition. The role of smartphones in IoT is not only a camera sensor but also a management device.

3. Materials and Methods

We performed recognition with 11 species of ornamental fishes, as shown in Fig. 1. The species names of the ornamental fishes are as follows.

(a) *Altolamprologus compressiceps*
(b) *Chilotilapia rhoadesii*
(c) *Cichlasoma* var. Kilin Parrot
(d) *Cyrtocara moorii* Albino
(e) *Heros severus*
(f) *Labeotropheus trewavasae* Chilumba
(g) *Labidochromis caeruleus*
(h) *Lethrinops* sp. red cap
(i) *Placidochromis* sp. Gold (Mbamba Bay)
(j) *Pseudotropheus socolofi*
(k) *Pterophyllum scalare* albino Manacapuru

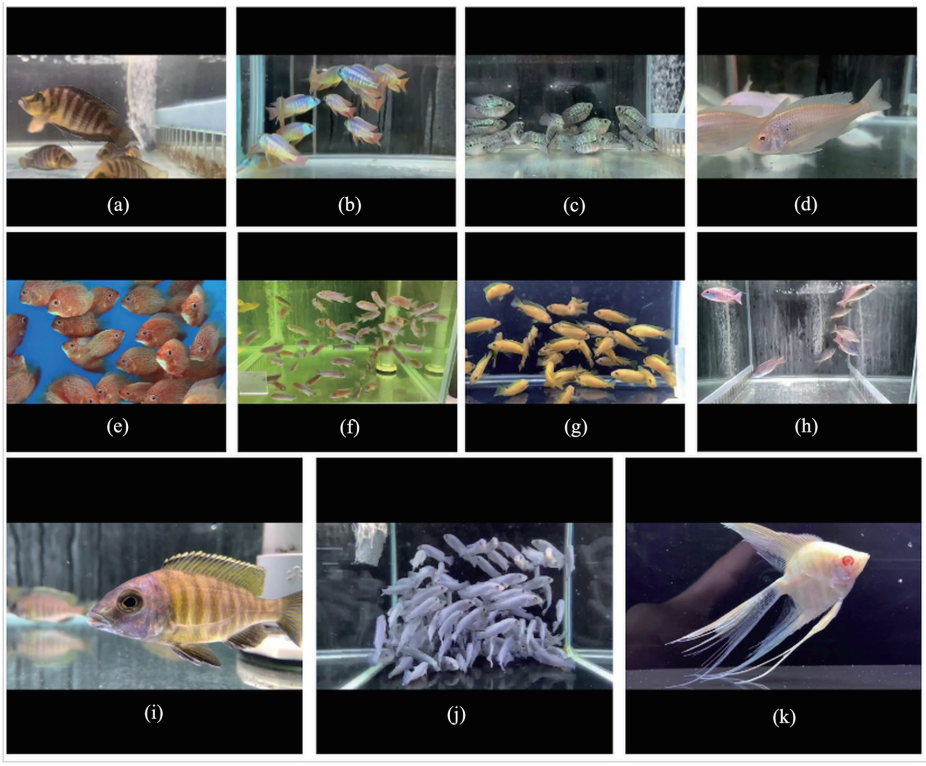

Fig. 1. (Color online) Names of ornamental fishes in order from left to right and from top to bottom are (a) *Altolamprologus compressiceps*, (b) *Chilotilapia rhoadesii*, (c) *Cichlasoma* var. Kilin Parrot, (d) *Cyrtocara moorii* Albino, (e) *Heros severus*, (f) *Labeotropheus trewavasae* Chilumba, (g) *Labidochromis caeruleus*, (h) *Lethrinops* sp. red cap, (i) *Placidochromis* sp. Gold (Mbamba Bay), (j) *Pseudotropheus socolofi*, and (k) *Pterophyllum scalare* albino Manacapuru.

The main configuration of the YOLOv3 model for ornamental fish recognition is listed as follows.

- Dataset: 2200 pictures
- Classes: 11
- Data augmentation: built-in (angle, saturation, exposure, hue)
- Train/test ratio: 80/20
- Initiate learning rate: 0.001
- Batch size: 24
- Image size: 608 × 608 (RBG)
- Iterations: 6000
- Pretrained weights on ImageNet for the convolutional layers: darknet53.conv.74

The YOLOv3 model was deployed on the public cloud AWS. The type of AWS EC2 instance is g4dn.4xlarge. It consists of an NVIDIA T4 Tensor Core GPU, each with 320 Turing Tensor cores, 2560 CUDA cores, and 16 GB of graphics memory, and 16 vCPU cores and 64 GB of main memory.

We adopted the microservice architecture to communicate each component loosely and flexibly. Table 2 shows the details and implementers of the services. The proposed ornamental fish recognition application was divided into five dependent microservices. User Interface (UI) service provided five tasks: take the photo, compress the photo, upload the photo via RESTful API, parse the response, and display the result. Vue.js is a good choice to implement these tasks and to deploy into mobile devices. RESTful API is a powerful interface for IoT devices and cloud computing. RESTful service not only communicates with IoT devices but also interacts with the database and YOLOv3 object detection model. Django is a high-level Python web framework that provides a *clean and pragmatic design* to developers to develop *web applications* rapidly. It enabled us to interact with other components easily. The database service stores the information of ornamental fishes. The recognition service uses the YOLOv3 model to recognize the objects and find the locations. We can query more information from the database service. This ornamental fish recognition application also draws the boundary box of the target fish,

Table 2
Microservices for ornamental fish recognition application.

Services	Implementer		Tasks
	Software	Hardware	
UI service	Vue.js	Mobile devices/ IoT	• Take photo • Compress/resize the photo • Call the RESTful API to upload photo • Parse the response from web server • Display the result
RESTful service	Django/Python	AWS EC2	• Handle RESTful API • Interact with database • Interact with YOLOv3 model
Database service	MySQL	AWS EC2	• Manipulate information of ornamental fishes
Image service	Nginx	AWS EC2	• Provide the processed images
Recognition service	YOLOv3 model	AWS EC2	• Train the dataset • Test/recognize the fish

assigns the species name, and predicts the probability of the target fish on the image being that species. The image service provides the space to store the images and allows the users to access the images that have been recognized.

Table 3 lists all the software and hardware elements of the ornamental fish recognition application. The UI service of this application installed in the mobile devices provides the user interface. The UI framework is used to deploy the application in various mobile phones, including Android and iPhone. Developers used HBuilderX IDE to write the related code with Vue.js on a desktop computer. They tested RESTful API via Advanced REST client to confirm the functionality. At the same time, deep learning team members started to train the YOLOv3 model on the cloud. Amazon AWS Elastic Compute Cloud (EC2) G4 instances are the industry's most cost-effective and versatile GPU instances for deploying machine learning models such as image classification, object detection, and speech recognition. G4dn instances feature NVIDIA T4 GPUs and custom Intel Cascade Lake CPUs, and are optimized for machine learning inference and small-scale training. These instances are also ideal for customers who prefer to use NVIDIA software and libraries such as CUDA, CuDNN, and NVENC.

4. Proposed Method

Figure 2 shows the architecture of the ornamental fish recognition application. It depicts the relation of each component. Table 4 defines the specifications of RESTful web API for uploading the image file from the client such as mobile devices to the cloud. The mobile device invokes the RESTful web API via HTTP POST request with URI. [AWSEC2] means the AWS EC2 public IP address. The request data via RESTful web API should be converted into multipart/form-data format, and the response data should be in JavaScript Object Notation (JSON) format. The request data consist of file names, image files, and image types. After the series process, the

Table 3
IoT device and cloud computing elements of ornamental fish recognition application.

Hardware	Software	Description
Mobile device	Hypertext Markup Language, cascading style sheets, JavaScript	Web technologies that provide a user interface (UI)
	UI framework	Open-source Vue.js toolkit to cooperate with mobile camera and album, compress the photo, and show the result
	Android/iOS	Mobile operating system
Desktop PC (macOS/ Windows)	Hbuilder X Editor	Front-end Web integrating development environment, to build mobile application on personal computer
	Advanced REST client	Help developers create and test custom HTTP requests such as RESTful API
AWS EC2 (g4dn.4xlarge)	YOLOv3 model	One-stage object detection model
	Django framework	A high-level Python Web framework to receive the photo, cooperate with YOLO, and provide the result.
	Nginx web server	A powerful web server to provide access to processed images

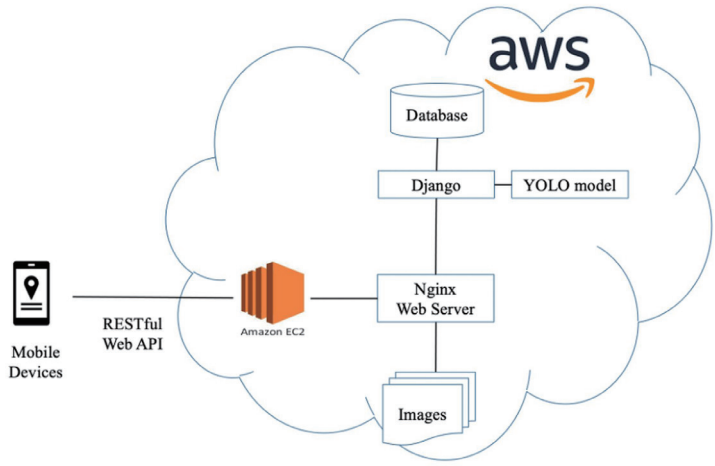

Fig. 2. (Color online) Architecture of ornamental fish recognition application.

Table 4
RESTful web API specifications for uploading image.

Description	Upload image file		
URI	http://[AWSEC2]/imgUpload/		
Method	POST	I/O format	I: multipart/form-data O: JSON
	Parameter	Type	Description
Input	fileUpload	String(100)	Name of image file
Output (array)	fishQtn	Int	Number of fish
	fishName	String(100)	Common name of fish
	distribution	Memo	Habits and habitat of fish
	latinName	String(100)	Latin species name of fish

RESTful web API will return the array, including the numbers, common names, habits and habitats, and species names of all the recognized fishes.

Figure 3 describes the flow of our proposed application through the swim lane diagram. The user selects the photo from the camera or album. The UI service of the mobile device resizes the photo into 1024 pixels in weight. It is about 10 times smaller than the original photo. This action does not affect the accuracy of object detection and indeed considerably shortens the transmission time. A mobile device sends the custom RESTful web API request and Nginx web server handles this HTTP request. In accordance with the configuration of Nginx, it forwards the HTTP request to Django via a fast CGI module. The custom RESTful web API writes this photo into a local image file and triggers the YOLOv3 model to recognize this image. The YOLOv3 object detection model anchors the boxes of these objects in this original image, predicts these objects, and finally returns the predicted result as a list array. The resulting array includes predicted fish, species name, probability, and location for each object. Django sends the predicted name to the database service to obtain more detailed information. Django decodes the result in the JSON format as a RESTful web API response. The mobile device receives the response message in JSON format, parses the message to gain the processed image of fish recognition, and finally shows the processed image on its screen.

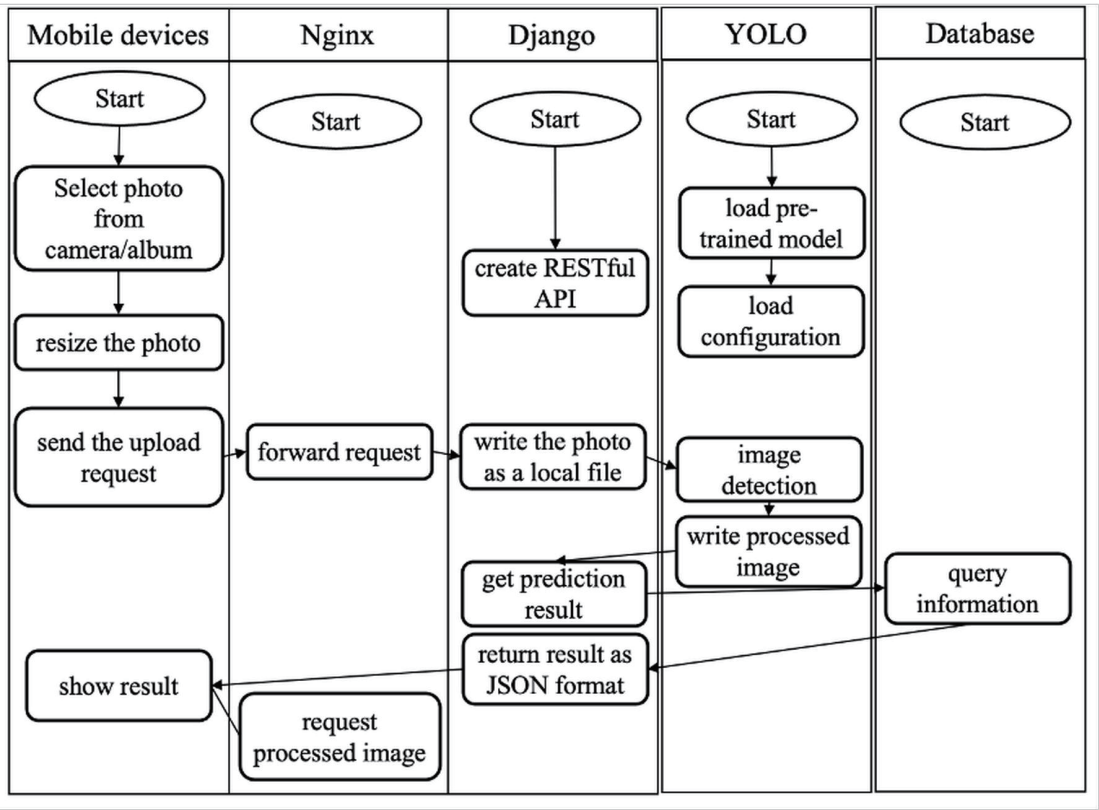

Fig. 3. Swimlane diagram for ornamental fish recognition application.

Fig. 4. (Color online) Forwarding process from Nginx web server to Django framework.

Figure 4 shows the forwarding process from the Nginx web server to the Django framework. The Nginx web server is listening on port 80 as a default web server. It receives all HTTP requests and separates the traffic by URI. uWSGI is named after the Web Server Gateway Interface (WSGI) and is used for connecting Python applications with web servers. uWSGI establishes a socket channel and Nginx communicates with uWSGI through the socket. uWSGI executes the Django code through the ini configuration file.

5. Experimental Results and Discussion

The top of Fig. 5 shows the result screen on our mobile device for the proposed ornamental fish recognition application. The middle part of Fig. 5 shows a part of the code for inference time. The first line specifies the name of the image to be inferred, the second line starts the

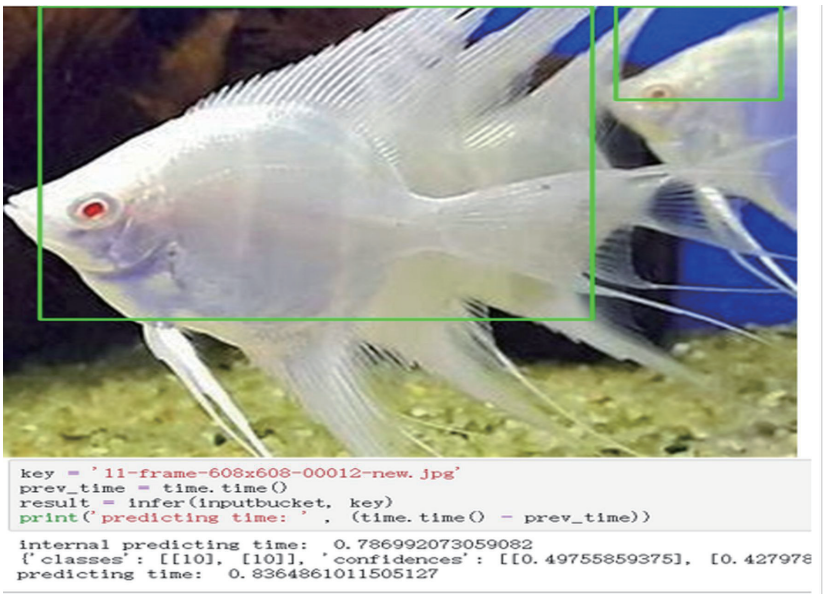

Fig. 5. (Color online) Result screen on mobile device for ornamental fish recognition application.

timing, the third line is for fish image recognition, and the fourth line shows the inference time. The bottom of Fig. 5 shows the inference time. The 'internal predicting time', 0.786 s, means the actual inference time in the YOLOv3 model, and the 'predicting time', 0.836 s, includes the time for plotting a boundary box for each recognized object. The original photo taken by the mobile devices was about 10.5 MB. After resizing the photo, the actual uploaded photo was 16 KB. The transfer volume is reduced by 660 times.

The results of our proposed application provided users with a convenient solution that allows them to use their mobile devices to identify 11 types of ornamental fish and see the information easily. The application development employed front-end programmers, cloud computing developers, and AI designers. The microservice architecture lets them work independently and simultaneously. These developers focused only on their own tasks. RESTful API and function calls connected each service.

6. Conclusions

In this study, we applied IoT, cloud computing management, object detection, and RESTful API methods to the proposed application. The proposed cost-effective and flexible architecture increases the practicality of the ornamental fish recognition system. This system exhibits satisfactory speed, completing the recognition in one second. Therefore, we suggested that more train datasets should be used for different species of fishes to keep the recognition accuracy high. This proposed application uses a microservice architecture, so it has a high degree of flexibility and can easily replace any service. Users can use a mobile phone with a camera sensor or any device that has a camera sensor and can invoke RESTful web API to use this image recognition service. However, this architecture cannot maintain the availability or reliability of these services.

Acknowledgments

Special thanks go to Creative Way Global Corporation for providing us with different species of fish and for financial support. This work was supported by the Excellent Young Key Teachers of Jiangsu University "Qing Lan Project" in 2020, the Teaching Innovation Team of Changzhou College of Information Technology, the Software Technology Specialty Group Construction Project of High Level Vocational Schools of the Ministry of Education of China, the Jiangsu Excellent Scientific and Technological Innovation Team Plan for Industrial Network and Industrial Big Data Application, the Smart Education Engineering Center under Grant no. CXPT201701G, and the Special Project of New Engineering Research and Practice under Grant no. 2018XGK001.

References

1 S.-B. Satam, N.-H. Sawant, M.-M. Ghughuskar, V.-D. Sahastrabuddhe, V.-V. Naik, A.-U. Pagarkar, and A.-N. Sawant: Adv. Agric. Res. Technol. J. **2** (2018) 193. http://isasat.org/vol-ii-issue-ii-july-2018/11_AARJ_II_2_Satam%20et%20al_193-197.pdf
2 K. Raja, P. Aanand, S. Padmavathy, and J.-S. Sampathkumar: Int. J. Fisheries Aquatic Stud. **7** (2019) 6. https://www.fisheriesjournal.com/archives/2019/vol7issue2/PartA/7-1-48-550.pdf
3 M. Solahudin, W. Slamet, and A.-S. Dwi: E&ES **147** (2018) 012014. http://doi.org/10.1088/1755-1315/147/1/012014
4 M. Kalske: Transforming Monolithic Architecture Towards Microservice Architecture (University of Helsinki, 2017). https://helda.helsinki.fi/bitstream/handle/10138/234239/transforming-monolithic-architecture.pdf
5 F. Ponce, G. Marquez, and H. Astudillo: Proc. 2019 IEEE 38th Int. Conf. Chilean Computer Science Society (IEEE, 2019) 1–7. https://doi.org/10.1109/SCCC49216.2019.8966423
6 L. De Lauretis: Proc. 2019 IEEE 30th Int. Conf. Software Reliability Engineering Workshops (IEEE, 2019) 93–96. https://doi.org/10.1109/ISSREW.2019.00050
7 J.-R.-C. Santos: An Effective and Effcient Web Platform for Monitoring, Control, and Management of Drones Supported by A New Microservices Approach (Iscte - Instituto Universit ario de Lisboa, 2020) Chap. 2. https://repositorio.iscte-iul.pt/bitstream/10071/22060/1/Master_Jorge_Cunha_Santos.pdf
8 B. Zion: Comput. Electron. Agric. **88** (2012) 125. https://doi.org/10.1016/j.compag.2012.07.010
9 Automatic Fish Counting in Aquariums: https://fenix.tecnico.ulisboa.pt/downloadFile/1689244997255462/resumo_65359.pdf (accessed April 2020).
10 P.-F. Newbury, P.-F. Culverhouse, and D.-A. Pilgrim: Aquaculture **133** (1995) 45. https://doi.org/10.1016/0044-8486(95)00003-K
11 C.-H. Huang: A Simple Design of Automatic Counting System for Fish Larvae (National Sun Yat-sen University, Kaohsiung, 2002) Chap. 2. https://hdl.handle.net/11296/6erp85
12 K.-D. Friedland, D. Ama-Abasi, M. Manning, L. Clarke, G. Kligys, and R.-C. Chambers: J. Sea Res. **54** (2005) 307. https://doi.org/10.1016/j.seares.2005.06.002
13 A. Flores, P. Crisostomo, and J. Lopez: Proc. 2008 IEEE 7th Int. Caribbean Conf. Devices, Circuits and Systems (IEEE, 2008) 1–4. https://doi.org/10.1109/ICCDCS.2008.4542660
14 W. Khantuwan and N. Khiripet: Proc. 2012 IEEE 9th Int. Conf. Electrical Engineering/Electronics, Computer, Telecommunications and Information Technology (IEEE, 2012) 1–4. https://doi.org/10.1109/ECTICon.2012.6254280
15 L. Coronel, W. Badoy, and C. Namoco: Int. Arab J. Inf. Tech. **15** (2018) 708. https://iajit.org/PDF/July%202018,%20No.%204/11042.pdf
16 J. Redmon, S. Divvala, R. Girshick, and A. Farhadi: Proc. 2016 IEEE Conf. Computer Vision and Pattern Recognition (IEEE, 2016) 779–788. https://doi.org/10.1109/CVPR.2016.91
17 J. Redmon and A. Farhadi: Proc. 2017 IEEE Conf. Computer Vision and Pattern Recognition (IEEE, 2017) 6517–6525. https://doi.org/10.1109/CVPR.2017.690
18 J. Redmon and A. Farhadi: arXiv preprint (2018). https://arxiv.org/abs/1804.02767
19 Y.-B. Lin and H.-C. Tseng: IEEE Access **7** (2019) 35457. https://doi.org/10.1109/ACCESS.2019.2905017

20 Y.-H. Liu, W.-H. Wu, X.-R. Huang, W.-H. Chen, and L.-B. Chen: Proc. 2021 IEEE Int. Conf. Consumer Electronics-Taiwan (IEEE, 2021) 1–2. https://doi.org/10.1109/ICCE-TW52618.2021.9603058
21 M. H. Rohit, Z. T. Hoque, S. M. Karim, and S. Siddique: Proc. 2019 IEEE/ACM Int. Conf. Software Engineering Research and Practices for the Internet of Things (IEEE, 2019) 49–52. https://doi.org/10.1109/SERP4IoT.2019.00015
22 B.-E. Agossou: IoT & AI Based System to improve Fish Farming: Case study of Benin (Kobe Institute of Computing, 2021). https://www.researchgate.net/profile/Bidossessi-Agossou-2/publication/354313538_IoT_AI_Based_System_to_improve_Fish_Farming_Case_study_of_Benin/links/6130b3b2c69a4e4879737294/IoT-AI-Based-System-to-improve-Fish-Farming-Case-study-of-Benin.pdf
23 A. Riansyah, R. Mardiati, M. R. Effendi, and N. Ismail: Proc. 2020 IEEE 6th Int. Conf. Wireless and Telematics (IEEE, 2020) 1–4. https://doi.org/10.1109/ICWT50448.2020.9243620
24 C.-T. Yeh and M.-S. Ling: Sens. Mater. **33** (2021) 3027. https://doi.org/10.18494/SAM.2021.3240
25 L.-D. Van, Y.-B. Lin, T.-H. Wu, Y.-W. Lin, S.-R. Peng, L.-H. Kao, and C.-H. Chang: Sensors **19** (2019) 1763. https://doi.org/10.3390/s19081763

About the Authors

Chi-Tsai Yeh received his B.S. degree from National Cheng Kung University in 1995, his M.S. degree from National Sun Yat-Sen University in 1997, and his Ph.D. degree from National Kaohsiung University of Science and Technology in 2019. From 2011 to 2019, he was an assistant professor at Shih Chien University. Since 2019, he has been an associate professor at Changzhou College of Information Technology. His research interests include computing vision, cloud computing, and sensors. (yehchitsai@czcit.edu.cn)

Tzuo-Ming Chen received his B.S. degree in Hydraulic Engineering from Chuang Yuan Christian University in 1982, his M.S. degree in Computer Science from Bradley University in 1987, and his Ph.D. degree in Computer Science from Illinois Institute of Technology, U.S.A, in 1995. He is an Assistant Professor with the Department of Information Technology and Communication, Shih Chien University. His research interests include network construction, cloud computing, and software-defined network. (jchen@g2.usc.edu.tw)

Zhong-Jie Liu received his B.S. degree from Xinyang Normal University in 2006 and his M.E. degree from Changzhou University in 2010. From 2010 to 2015, he worked in Changzhou Institute of Advanced Manufacturing Technology, engaged in intelligent control systems, software architecture design and image recognition, and other fields of research and development. In 2015, he joined Changzhou College of Information Technology. His research interests include robot control system, machine learning, and image processing. (156247874@qq.com)