Hindawi
Applied Computational Intelligence and Soft Computing
Volume 2020, Article ID 3738108, 13 pages
https://doi.org/10.1155/2020/3738108

Research Article

Fish Detection Using Deep Learning

Suxia Cui,[1] **Yu Zhou**,[1] **Yonghui Wang**,[2] **and Lujun Zhai**[1]

[1]*Department of Electrical and Computer Engineering, Prairie View A&M University, Prairie View, TX 77446, USA*
[2]*Department of Computer Science, Prairie View A&M University, Prairie View, TX 77446, USA*

Correspondence should be addressed to Suxia Cui; sucui@pvamu.edu

Received 10 July 2019; Revised 25 September 2019; Accepted 4 November 2019; Published 23 January 2020

Academic Editor: Aniello Minutolo

Recently, human being's curiosity has been expanded from the land to the sky and the sea. Besides sending people to explore the ocean and outer space, robots are designed for some tasks dangerous for living creatures. Take the ocean exploration for an example. There are many projects or competitions on the design of Autonomous Underwater Vehicle (AUV) which attracted many interests. Authors of this article have learned the necessity of platform upgrade from a previous AUV design project, and would like to share the experience of one task extension in the area of fish detection. Because most of the embedded systems have been improved by fast growing computing and sensing technologies, which makes them possible to incorporate more and more complicated algorithms. In an AUV, after acquiring surrounding information from sensors, how to perceive and analyse corresponding information for better judgement is one of the challenges. The processing procedure can mimic human being's learning routines. An advanced system with more computing power can facilitate deep learning feature, which exploit many neural network algorithms to simulate human brains. In this paper, a convolutional neural network (CNN) based fish detection method was proposed. The training data set was collected from the Gulf of Mexico by a digital camera. To fit into this unique need, three optimization approaches were applied to the CNN: data augmentation, network simplification, and training process speed up. Data augmentation transformation provided more learning samples; the network was simplified to accommodate the artificial neural network; the training process speed up is introduced to make the training process more time efficient. Experimental results showed that the proposed model is promising, and has the potential to be extended to other underwear objects.

1. Introduction

The ocean is full of mystery and the underwater exploration has always been an exciting topic. Nowadays, robotics has been widely adopted into our daily lives. The AUV is one type of robot, which is gaining more and more attention [1, 2]. It must be equipped with a sophisticate onboard computer, Inertial Measurement Unit (IMU), and other sensors to be able to support a preprogrammed navigation system [1]. Authors have experience on design and function of an AUV [3, 4] for competitions. The AUV, as shown in Figure 1, is featured with an i7-based industrial motherboard plus an ARM microcontroller. Detail hardware layout and mechanical balancing scheme are introduced in [3, 4]. It passed the qualification and became one of the eleven finalists at the 2017 IEEE Singapore AUV Challenge [5]. This competition was hosted in a swimming pool of clear water. The tasks did not need a high-resolution camera, so the major processor was not chosen to be of high performance. After this the AUV retired from the competition, authors realized it was time to revise the system to conquer real life tasks. As of now, most of the robot control platforms were shifting to Systems-On-Chip (SOC) [6, 7]. To move forward and add more functionalities to the AUV, one goal is to switch from a clear swimming pool environment to a real ocean water condition. Therefore, the hardware has to be upgraded to high resolution digital camera along with a powerful onboard computer, such as NVIDIA JETSON AGX XAVIER development board. So, before upgrading the whole system with integrated vision, research on an off-line simulation of the computer vision module was conducted. Fishes of many kinds were chosen to be the objects to build up the training and testing data set. Ocean water conditions vary from place to place. In the Gulf of Mexico where the authors reside, the water is not as clear as in the east or west coast of the United States. Thus, how to identify fish from the blurred sea water is most challenging in this research. One of the

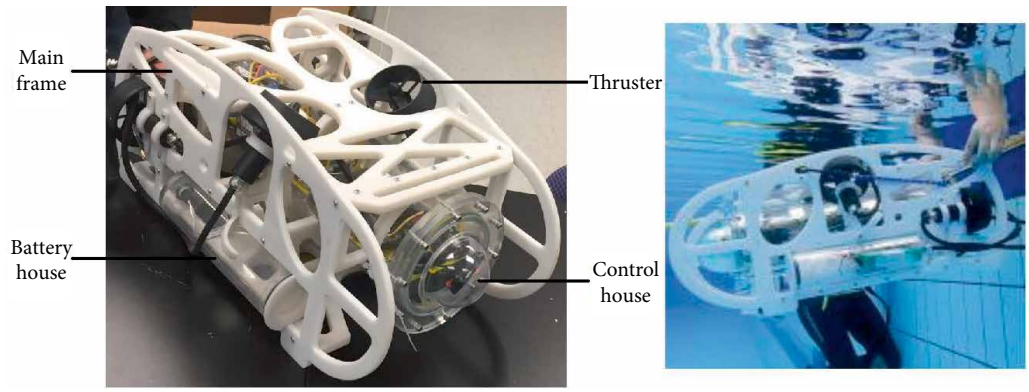


Figure 1: AUV and competition environment.

solutions is to adopt ultrasonic technology [8, 9]. To some extent, it was proved to be effective for the fish industry where a rough quantity of fish is sufficient enough. However, because of low resolution, it is difficult to differentiate objects in a complex environment that has mixed fishes, turtles, etc. The goal of this research is to investigate the object detection scheme under real sea water through an AUV build-in digital camera. Researchers have successfully adopted the digital camera as a tool for capturing images from the ocean to improve underwater robot vision [10], but the vehicle was remotely operated (ROV) instead of an AUV.

2. Literature Review

The main contribution of this research is to introduce deep learning methodology to accomplish fish identification in blurry ocean water. As a result, the approach improved computer vision into an AUV system through an applicable neural network.

2.1. Computer Vision. Computer vision uses computers with imaging sensors to imitate human visual functions that extract features from obtained data set, analyse and classify them to assist in decision making. It usually involves many fields of knowledge such as high-level computer programming, image processing, artificial intelligence (AI), and so on. For example, manufacture industry uses it to check the defection or improve the quality from large quantities of products [11, 12]. There are mature applications on face detection and emotion observation at the airport and other security checking points [13–15]. Medical doctors' use certain diagnose software to assist in identifying tumours and other abnormal tissues from medical imaging [16]. The agricultural industry adopts the computer vision to decision making system for predicting the yield from the field [17]. Google is designing its own self-driving car with a visual range of about 328 feet and the car can recognize traffic signs and avoid pedestrians [18]. Many state-of-the-art examples indicate that computer vision is changing our daily lives. To improve the performance, besides traditional image processing skills, deep learning algorithms which imitate our brain are widely adopted.

2.2. Deep Learning. The concepts of deep learning with neural network has arisen decades ago. It was originally developed by researcher LeCun et al. in 1998 [19]. He designed a five-layer classifier named LeNet5 using a Convolutional Neural Network (CNN). Due to dramatic improvement in computing power and the explosion of big data, deep learning is able to make tremendous achievements in the past several years. Deep learning is based on big data collected in a certain field. Learning resources from massive data are extremely important. Deep means that a neural network has lots of layers for imitating our brain. With the advent of high-performance GPU, ASIC accelerators, cloud storage, and powerful computing facility, it is now possible to collect, manage, and analyse big data sets. Because only with data sets large enough, can overfitting problems be solved in deep learning. And the enhanced computing power can accelerate the speed of time-consuming training process.

Deep leaning based approaches are increasingly applied in many fields, and have significant advantage over traditional algorithms in computer vision and object detection. The performance of many robotics systems has been improved by incorporating deep learning. Take Google's AlphaGo as an example, it studied human's learning behavior and in return compete with the famous Go player [20].

To be able to foster deep learning in computer vision, enough examples from images collected beforehand is critical. ImageNet is a good example [21]. One contribution of this research includes developing a database of fish in ocean water to support training and testing. Nevertheless, a learning algorithm is important as well. Traditional computer vision and image processing approaches suffered from the accuracy of feature extraction, while deep learning method can be utilized to improve the technique through neural network.

2.3. Neural Network. Over the past few years, neural networks in deep learning were getting increasingly popular. In 2012, researcher Krizhevsky et al. adopted CNN to accomplish images classification in the ImageNet Large Scale Visual Recognition Challenge [22, 23], and the test accuracy was significantly higher than traditional algorithms. Due to this achievement, the interest in deep learning with neural network has been raised [24]. In 2014, Ross et al. proposed an algorithm called Fast R-CNN which aims to convert object

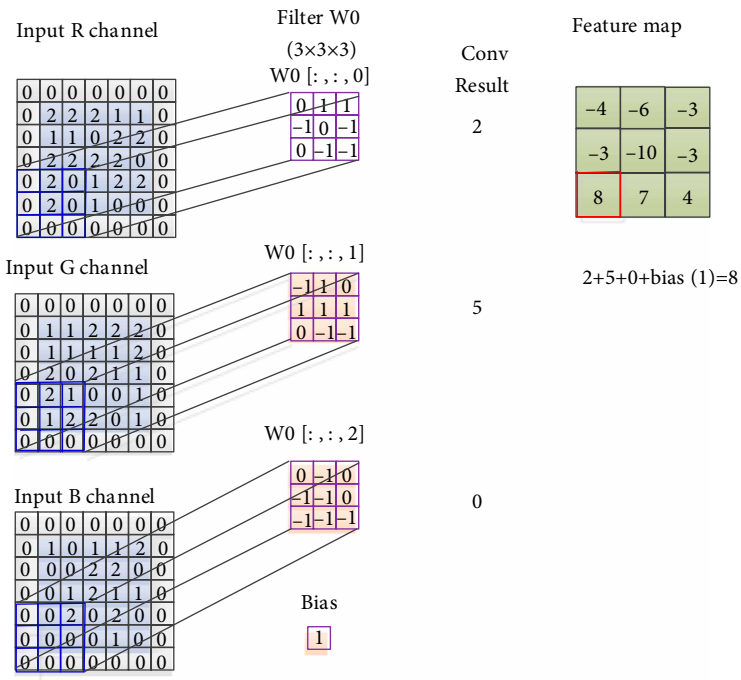


Figure 2: Convolutional operation on a RGB color image [30].

identification into a regression problem [25]. The mean average precision was improved by almost 30% compared to the previous best result 53.3% on ImageNet Large Scale Visual Recognition Challenge in 2012. The amount of calculation was massive because features from different sizes of thousands of proposals in each image would be extracted. Since Faster R-CNN reduced the computational burden dramatically, it has been widely adopted recently in computer vision which involves target detection, image classification, and object identification. YOLO proposed in Facebook is also a milestone for corresponding research [26, 27].

3. Materials and Methods

In this paper, a CNN model with image segmentation is introduced for fish detection from in blurry ocean water. Specific data set was developed to support this research. The data augmentation transformation scheme was adopted to obtain more learning resources because the original images in the particular environment are not sufficient for training purpose. To solve the overfitting problem, the dropout algorithm is applied. Because our goal is to incorporate this system into an AUV which requires real-time applications, some trade-offs were discussed to reduce processing time. In this section, detail system design with optimization approaches is addressed.

3.1. CNN Architecture. A CNN model usually consists of many layers, such as an input layer, convolutional layers with nonlinear units, and fully connected layers [28, 29]. An example of CNN is demonstrated in Figure 2. The first layer is the input layer which receives image information as learning resources from the outside world. The following layers are convolutional layers, which are responsible for extracting features from images. Convolution operation is one of the common mathematical operations. The convolution formula of two discrete functions is shown in Equation (1):

$$(f * g)[n] = \sum_{m=-\infty}^{\infty} f[m]g[n-m]. \tag{1}$$

The data set consists of 256 levels of RGB color images. The 3×3 matrix, W0 below, is called a kernel or a filter. In practice, convolutional operations are performed on R, G, and B channels respectively, and then the results are summed up to obtain each element in the feature map as shown in Figure 2.

In order to extract the features of an object more accurately, a lot of filters are used in each convolutional layer. For example, to extract features, such as edges, texture, etc., corresponding filters are available as shown in Figure 3.

When performing the convolutional operation, the size of the feature map is under consideration. There are three main factors that influence its size: depth, stride, and padding. Figure 4 illustrates the feature map where depth is 3, stride is 1, and with zero padding.

For a complex neural network, usually there are two types of connections between two adjacent layers. They are the fully connected layer and locally connected neural layer respectively as illustrated in Figure 5. For a fully connected neural net, all pixels in the input layer are connected with each neuron in the hidden layer as shown in Figure 5(a). It is common that the last two layers in a CNN are fully connected layers. They are the softmax and output layer, respectively. Because a huge number of parameters will increase the amount of computation and delay the processing. For a locally connected neural network, only a portion of pixels in the input layer are connected with the following neuron in the hidden layer as shown

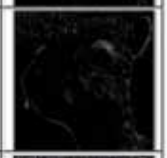
Operation	Filter	Feature map
Identity	$\begin{pmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{pmatrix}$	
Edge detection	$\begin{pmatrix} 1 & 0 & -1 \\ 0 & 0 & 0 \\ -1 & 0 & 1 \end{pmatrix}$	
	$\begin{pmatrix} 1 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{pmatrix}$	
	$\begin{pmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{pmatrix}$	
Sharpen	$\begin{pmatrix} 0 & -1 & 0 \\ -1 & 5 & -1 \\ 0 & -1 & 0 \end{pmatrix}$	
Box blur	$\frac{1}{9}\begin{pmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{pmatrix}$	
Guassian blur	$\frac{1}{16}\begin{pmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 1 & 1 \end{pmatrix}$	

Figure 3: Convolution operation application in image feature exaction [31].

in Figure 5(b). This type of connection will reduce the number of connections and speed up the system.

The Convolutional layer in CNN uses local connections as shown in Figure 6. For example, the value-8 in the feature map is only connected with a 3×3 matrix [0, 0, 0; 0, 1, 1; 0, 1, 2] from input image and has nothing to do with the remaining parts of the input image pixels.

The parameters for fully or local connectivity for all the layers in this CNN are listed in Table 1.

The system can be visualized with simplification in Figure 7.

3.2. System Validation Using ImageNET Dataset. Before applying this system into the ocean fish data set developed in this research, authors downloaded images from the well-known ImageNet ILSVRC [21] to do a system validation testing through object classification. There are 500 images with 20 classes ranging from fish, coral, sea turtle, frog, ship, etc. Here all the RGB images are rescaled to 448×448. Ground

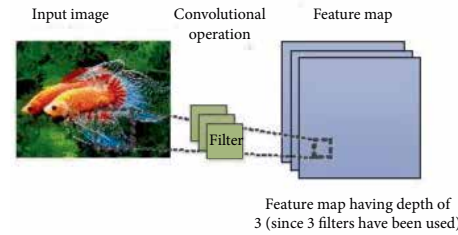


Figure 4: Feature extractor using convolutional operation.

truth images are obtained from operate LabelImg software manually. Each image is divided into a grid of 7×7 cells. Each cell will predict the two bounding boxes location information and class information made up of a $1 \times 1 \times 30$ vector. This vector consists of object centre coordinates (X, Y), the width ω, and height h of the bounding box confidence scores, and predicted probabilities of fish, as shown in Figure 8.

To predict the target location of an image, the target is displayed in a bounding box. There are always errors between the ground truth and the predictions. Loss function was developed to measure errors consisting of three parts: coordinate error, (Intersection over union) IoU error, and class error. Equation (2) gives the mathematics form of the loss function.

$$Loss = \sum_{i=0}^{S^2} coordError + IoUError + classError. \quad (2)$$

Here, IoU is used to measure position accuracy as shown in Figure 9.

Each grid cell in an image will predict K bounding boxes that encloses an object to predict the object localization and class. In addition, there is a confidence with each bounding box. Confidence score has nothing to do with the class of object. It just depicts how certain it is that the predicted box actually encloses the real object.

$$\text{Confidence} = \Pr(\text{object}) \times \text{IoU}, \quad (3)$$

where Pr(object) represents the probability of the object of interest. If there is an object in the grid cell, the Pr(object) is 1; otherwise, it is 0.

Usually, loss function is in the form of the sum of squared errors as shown below [33]. It consists of three parts which are localization errors, confidence errors, and probabilities errors.

$$\begin{aligned} Loss = {} & \sum_{i=0}^{S^2} \sum_{j=0}^{B} \left[(x_i - \hat{x}_l)^2 + (y_i - \hat{y}_l)^2 + (w_i - \hat{w}_l)^2 + \left(h_i - \hat{h}_l\right)^2 \right] \\ & + \sum_{i=0}^{S^2} \sum_{j=0}^{B} \left(C_i - \hat{C}_i\right)^2 + \sum_{i=0}^{S^2} \sum_{C \in class} \left(P_i(c) - P_i(\hat{c})\right)^2, \end{aligned} \quad (4)$$

where x_i, y_i are the ground truth coordinates of objects center; w_i, h_i are the width and height of the ground truth bounding box; $\hat{x}_i$, $\hat{y}_i$ are the predicted coordinates of the objects center; $\hat{w}_i$, $\hat{h}_i$ are the width and height of predicted bounding box. Above Figure 10 shows one set of output with confidence values from different classes.

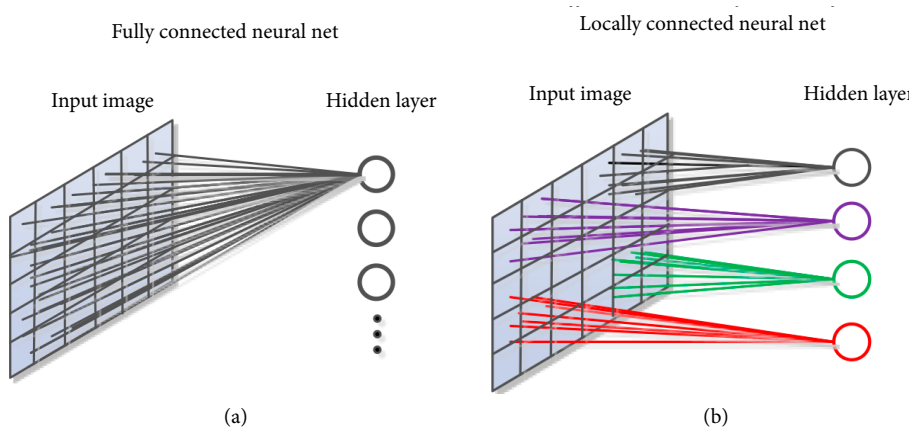


Figure 5: Fully connected neural net and locally connected neural net [32].

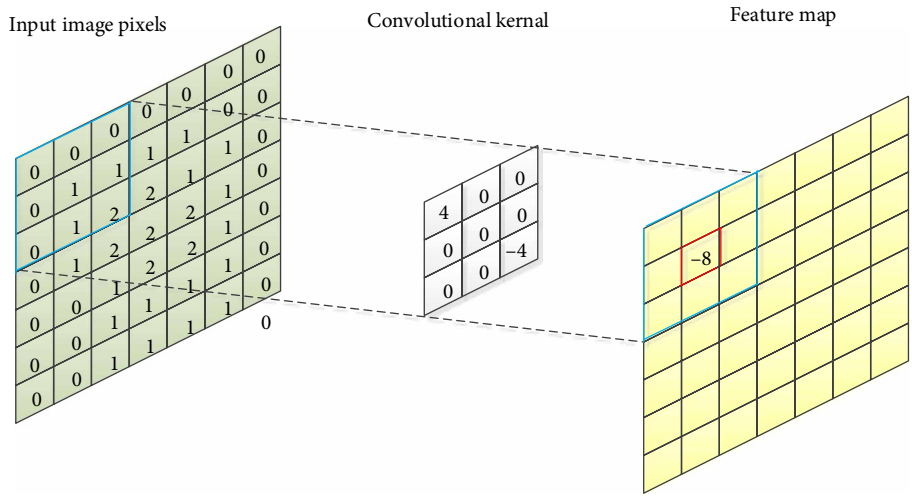


Figure 6: Convolutional operation using local connections.

3.3. Ground Truth Preparation for Real Ocean Environment. After testing the CNN system with perfect images without noise. Next step is to build our own dataset of fish in the ocean. Because it is difficult to obtain images from other kinds of objects such as the sea turtle, coral and so on. This part of the research, fish is the only object to be detected. For the collection of 410 images, many of them have multiple fish in one image, so the detection is challenging. The same method was chosen to create ground truth image. And all the parameters introduced before remain the same, only the class information is made up of a $1 \times 1 \times 18$ vector instead of $1 \times 1 \times 30$ because of reduce in the classes. Figure 11 illustrates one labelled image example. It is obvious that this data set is totally different from the ideal images from ImageNET.

4. Data Augmentation

Since deep learning is based on large training dataset for the system to learn and build up the identification knowledge, enough data has to be provided as learning resources to extract object features [34]. The lacking of data would bring overfitting problem. Images were collected in real underwater environment from the Gulf of Mexico and are going to be used to attract object features. However, the number of original images collected from a particular environment is not large enough to train the system. Therefore, data augmentation transformation was performed geometrically, which changed the pixels location while the images features remained unchanged as shown in Figure 12. Four types of data augmentation transformation were adopted, which doubled the number of original images to make the training dataset sufficient.

(1) Rotation: rotate the images by random angles;
(2) Scale: the images are scaled to different sizes according to scale factors set;
(3) Crop: crop patches of images;
(4) Mirror symmetry: flip the original images horizontally or vertically.

4.1. Dropout Algorithm. One of the common problems in deep learning is overfitting, which refers to the fact that the testing

Table 1: Parameters in CNN model with image segmentation.

Layer	Input size	Filter size	Stride	Output size
Conv. 1	$[N \times 448 \times 448 \times 3]$	$[7 \times 7 \times 3 \times 64]$	2	$[N \times 224 \times 224 \times 64]$
Maxpool 1	$[N \times 224 \times 224 \times 64]$	$[2 \times 2]$	2	$[N \times 112 \times 112 \times 32]$
Conv. 2	$[N \times 112 \times 112 \times 32]$	$[3 \times 3 \times 32 \times 192]$	1	$[N \times 112 \times 112 \times 192]$
Maxpool 2	$[N \times 112 \times 112 \times 192]$	$[2 \times 2]$	2	$[N \times 56 \times 56 \times 96]$
Conv. 3	$[N \times 56 \times 56 \times 96]$	$[3 \times 3 \times 128 \times 256]$	1	$[N \times 56 \times 56 \times 256]$
Conv. 4	$[N \times 56 \times 56 \times 128]$	$[3 \times 3 \times 128 \times 256]$	1	$[N \times 56 \times 56 \times 256]$
Conv. 5	$[N \times 56 \times 56 \times 256]$	$[1 \times 1 \times 256 \times 256]$	1	$[N \times 56 \times 56 \times 256]$
Conv. 6	$[N \times 56 \times 56 \times 256]$	$[3 \times 3 \times 256 \times 512]$	1	$[N \times 56 \times 56 \times 512]$
Maxpool 3	$[N \times 56 \times 56 \times 512]$	$[2 \times 2]$	2	$[N \times 28 \times 28 \times 256]$
Conv. 7	$[N \times 28 \times 28 \times 256]$	$[1 \times 1 \times 256 \times 256]$	1	$[N \times 28 \times 28 \times 256]$
Conv. 8	$[N \times 28 \times 28 \times 256]$	$[1 \times 1 \times 256 \times 256]$	1	$[N \times 28 \times 28 \times 256]$
Conv. 9	$[N \times 28 \times 28 \times 256]$	$[1 \times 1 \times 256 \times 256]$	1	$[N \times 28 \times 28 \times 256]$
Conv. 10	$[N \times 28 \times 28 \times 256]$	$[1 \times 1 \times 256 \times 256]$	1	$[N \times 28 \times 28 \times 256]$
Conv. 11	$[N \times 28 \times 28 \times 256]$	$[3 \times 3 \times 256 \times 512]$	1	$[N \times 28 \times 28 \times 512]$
Conv. 12	$[N \times 28 \times 28 \times 512]$	$[3 \times 3 \times 512 \times 512]$	1	$[N \times 28 \times 28 \times 512]$
Conv. 13	$[N \times 28 \times 28 \times 512]$	$[3 \times 3 \times 512 \times 512]$	1	$[N \times 28 \times 28 \times 512]$
Conv. 14	$[N \times 28 \times 28 \times 512]$	$[3 \times 3 \times 512 \times 512]$	1	$[N \times 28 \times 28 \times 512]$
Conv. 15	$[N \times 28 \times 28 \times 512]$	$[1 \times 1 \times 512 \times 512]$	1	$[N \times 28 \times 28 \times 512]$
Conv. 16	$[N \times 28 \times 28 \times 512]$	$[3 \times 3 \times 512 \times 1024]$	1	$[N \times 28 \times 28 \times 1024]$
Maxpool 4	$[N \times 28 \times 28 \times 1024]$	$[2 \times 2]$	2	$[N \times 14 \times 14 \times 512]$
Conv. 17	$[N \times 14 \times 14 \times 512]$	$[1 \times 1 \times 512 \times 256]$	1	$[N \times 14 \times 14 \times 256]$
Conv. 18	$[N \times 14 \times 14 \times 256]$	$[3 \times 3 \times 256 \times 1024]$	1	$[N \times 14 \times 14 \times 1024]$
Conv. 19	$[N \times 14 \times 14 \times 1024]$	$[1 \times 1 \times 1024 \times 512]$	1	$[N \times 14 \times 14 \times 512]$
Conv. 20	$[N \times 14 \times 14 \times 512]$	$[3 \times 3 \times 512 \times 1024]$	1	$[N \times 14 \times 14 \times 512]$
Conv. 21	$[N \times 14 \times 14 \times 512]$	$[3 \times 3 \times 512 \times 1024]$	1	$[N \times 14 \times 14 \times 1024]$
Conv. 22	$[N \times 14 \times 14 \times 1024]$	$[3 \times 3 \times 1024 \times 1024]$	1	$[N \times 7 \times 7 \times 1024]$
Conv. 23	$[N \times 7 \times 7 \times 1024]$	$[3 \times 3 \times 1024 \times 1024]$	2	$[N \times 7 \times 7 \times 1024]$
Conv. 24	$[N \times 7 \times 7 \times 1024]$	$[3 \times 3 \times 1024 \times 1024]$	1	$[N \times 7 \times 7 \times 1024]$
Fully conn.1	$[N \times 7 \times 7 \times 1024]$	$[1024 \times 4]$	Multi	$[1 \times 4096]$
Fully conn. 2	$[1 \times 4096]$	$[4096 \times 7 \times 7 \times 30]$	Multi	$[7 \times 7 \times 30]$

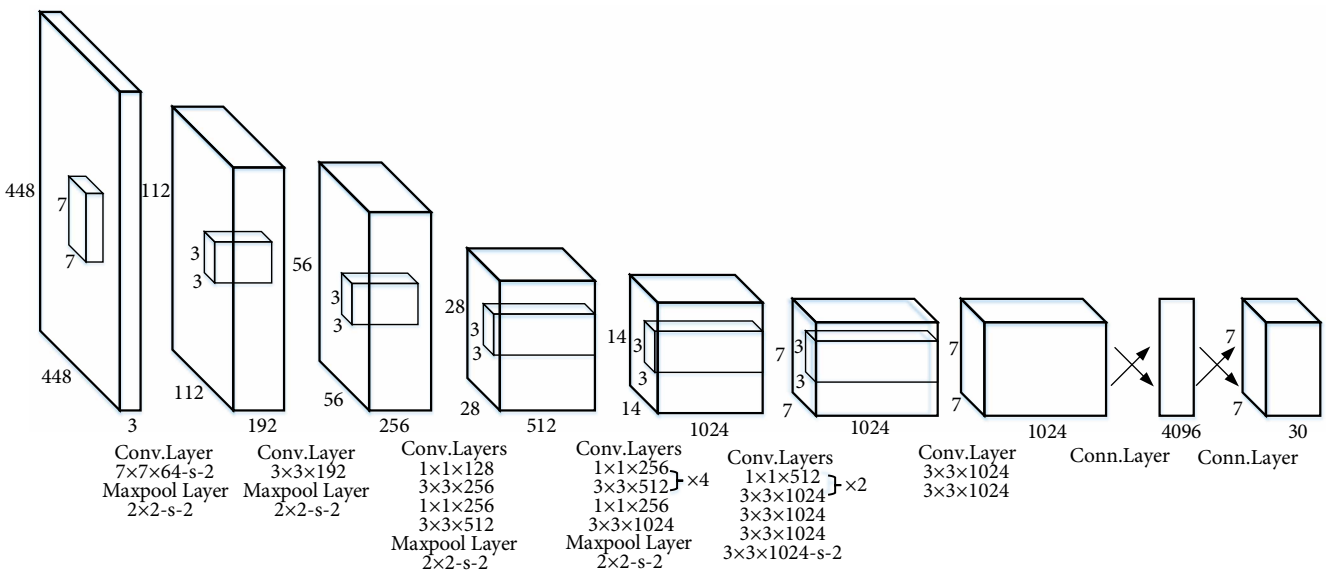


Figure 7: CNN architecture for object identification [26].

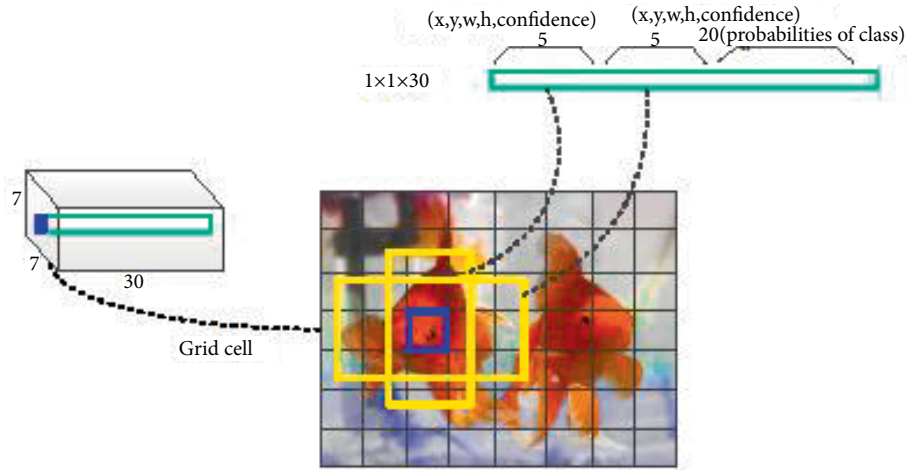


Figure 8: Output of CNN model using ImageNET.

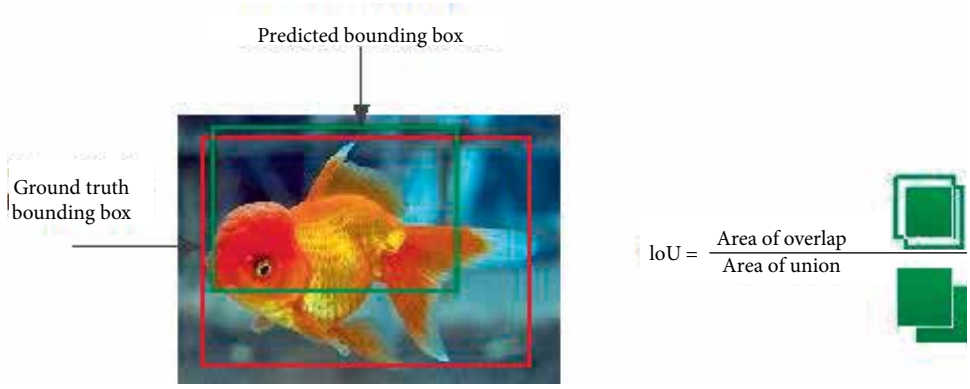


Figure 9: Intersection over union.

Figure 10: Object classification results from ImageNET data set.

accuracy is much lower than the training accuracy. In this case, a model with high performance feature is built using real world training data. If it turns out to have overfitting issue, the robustness of the model is worth consideration. Apart from the lacking of learning data, which will also cause overfitting problem, the huge amount of parameters in a neural network would lead to the overfitting issue. Therefore, the dropout algorithm [35] was introduced into the system to simplify the model, which is depicted in Figure 5.

Dropout means that we remove some nodes temporarily from the network according to the probability setting in the process of learning. In practice, some features can be extracted only when some hidden relationships exist, which decreased the robustness of the deep learning model. On the other hand,

Figure 11: Labelled ground truth.

dropout removes a hidden fixed relationship between nodes, anti-interference ability can be improved and overfitting problem can be solved to some extent. L1 and L2 regularization are

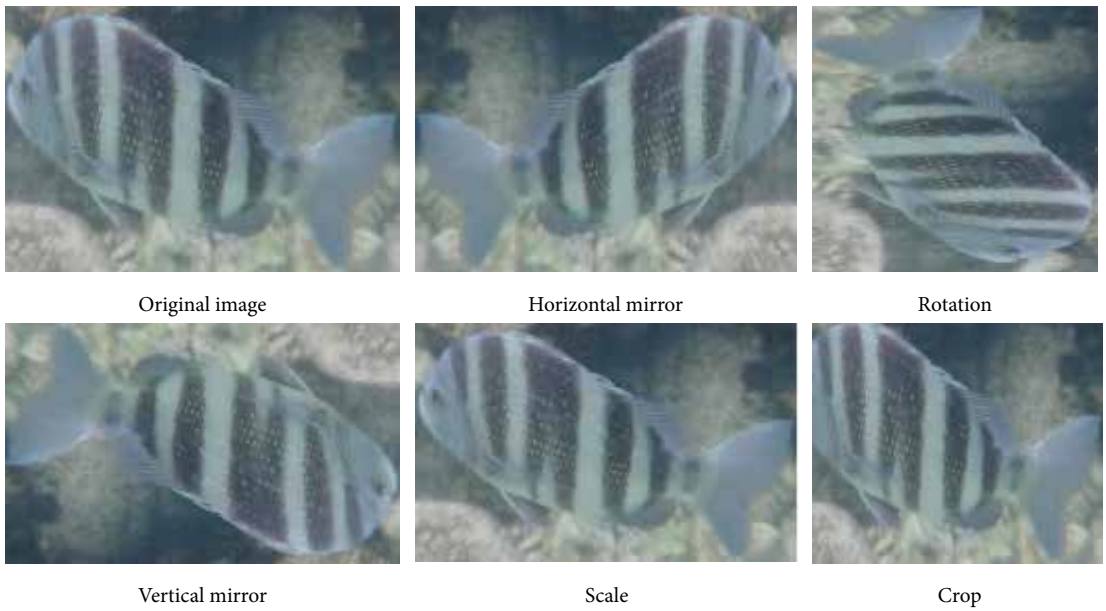


Figure 12: Data augmentation transformation.

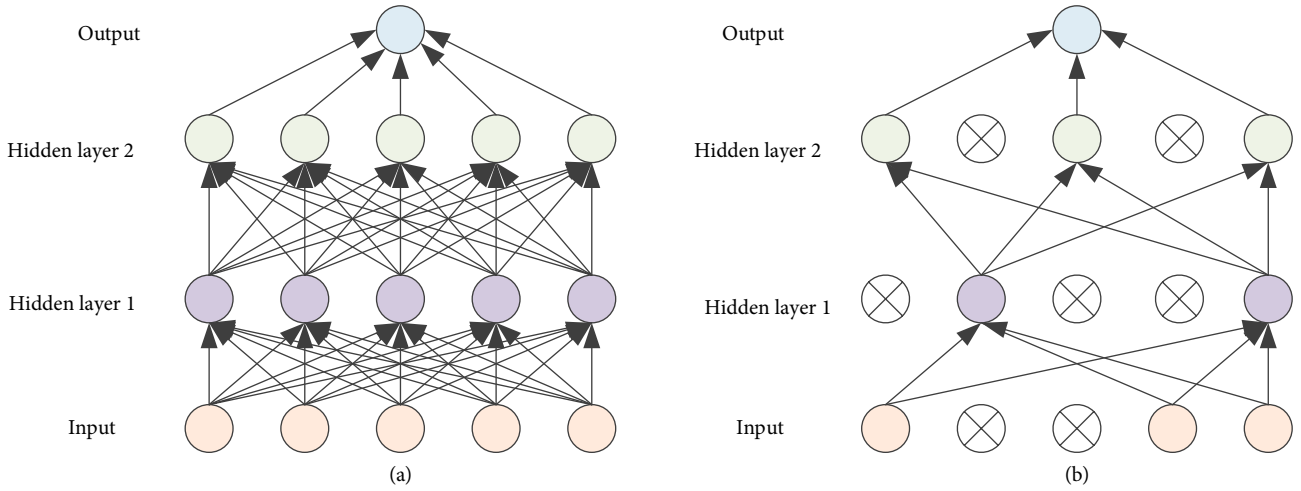


Figure 13: A standard neural network model (a) and a network model with dropout (b) [35].

achieved by modifying the cost function, while dropout is implemented by modifying the neural network itself, which is a technique used when training the network. In each iteration of the training process, authors randomly drop some of the neurons, and set the probability of eliminating nodes in the neural network for each layer of the network. For example, the value is set to 0.5, as shown in Figure 13 on the left. The neurons are discarded, then the connections from the node are removed, and finally a network with fewer nodes and smaller scales is obtained. The network structure during this training will be simplified to the one shown in Figure 13 on the right.

4.2. Refine Loss Function. YOLO improved loss function from Equations (4) and (5), [26]. Three coefficients were placed before the error terms in proportion to its contribution to the loss. As shown in Equation (5), the first two terms are related to coordinate the identified object with x and y to denote the object location, while w and h refer to the width and height of the bounding box. In order to have more weight in the first two terms, γ_{cord} was assigned to be the largest number, which had a value of 5. Thus, the weight of localization error got enhanced. In terms of IoU error computation, when the object center falls in this cell, the weight of IoU error should be increased in order to predict location accurately. The value of γ_{noobj} is set to be 0.5 to refine the IoU error. For the same error value, the effect of large object error on the detection should be less than the effect of small object error on the detection. This is because the same bias accounts for the proportion of large objects is much smaller than the proportion of the same deviation to small objects. Therefore, it is supposed to increase the contribution to loss due to bigger object IoU error. Square roots of width and height were chosen to replace their original forms. For same bias value, the square root error from the big box is smaller than the small one.

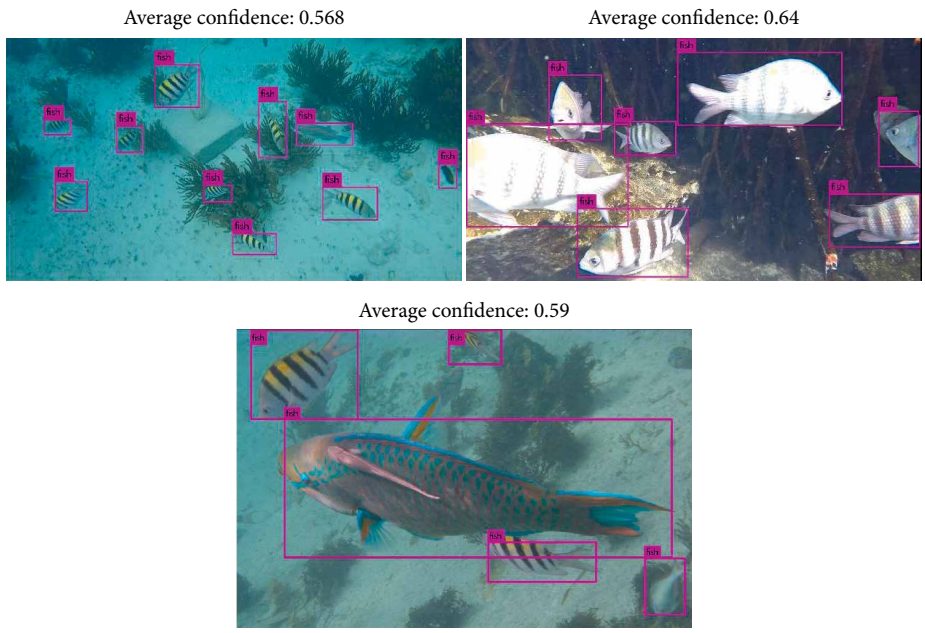


Figure 14: Enhancement of identification accuracy with data augmentation.

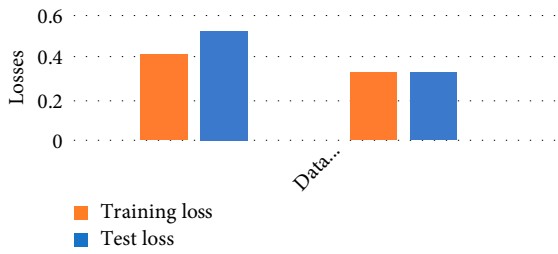


Figure 15: Comparison of training loss.

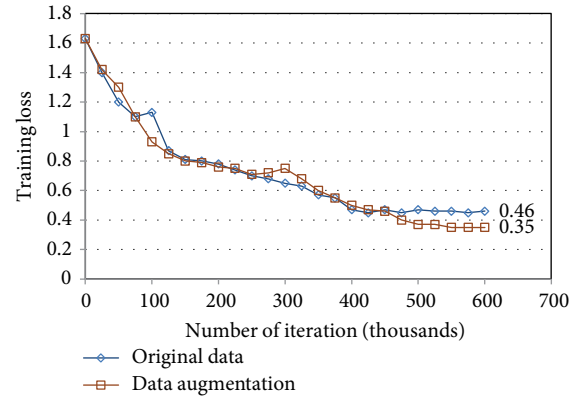


Figure 16: Effect of data augmentation on overfitting.

$$
\begin{aligned}
Loss =& \lambda_{coord} \sum_{i=0}^{S^2} \sum_{j=0}^{B} \left[1_{ij}^{obj} \left(x_i - \hat{x}_l \right)^2 + \left(y_i - \hat{y}_l \right)^2 \right] \\
&+ \lambda_{coord} \sum_{i=0}^{S^2} \sum_{j=0}^{B} 1_{ij}^{obj} \left[\left(\sqrt{w_i} - \sqrt{\hat{w}_l} \right)^2 + \left(\sqrt{h_i} - \sqrt{\hat{h}_l} \right)^2 \right] \\
&+ \sum_{i=0}^{S^2} \sum_{j=0}^{B} 1_{ij}^{obj} \left(C_i - \hat{C}_l \right)^2 + \lambda_{noobj} \sum_{i=0}^{S^2} \sum_{j=0}^{B} 1_{ij}^{nobj} \left(C_i - \hat{C}_l \right)^2 \\
&+ \sum_{i=0}^{S^2} 1_{ij}^{obj} \sum_{C \in class} \left(P_i(c) - P_l(\hat{c}) \right)^2.
\end{aligned} \tag{5}
$$

In this paper, authors refined the loss function to fit for multiple fish application. The proposed loss function is regularized to reduce the small dataset and overfitting problem, L2 regularization is to add a regularization function after the cost function which is listed in the Equations (6) and (7).

$$
\begin{aligned}
Loss =& \sum_{i=0}^{S^2} \left(coordError + IoUError + classError \right) \\
&+ \text{L2 Regularization},
\end{aligned} \tag{6}
$$

$$
\begin{aligned}
Loss =& \lambda_{coord} \sum_{i=0}^{S^2} \sum_{j=0}^{B} \left[1_{ij}^{obj} \left(x_i - \hat{x}_l \right)^2 + \left(y_i - \hat{y}_l \right)^2 \right] \\
&+ \lambda_{coord} \sum_{i=0}^{S^2} \sum_{j=0}^{B} 1_{ij}^{obj} \left[\left(\sqrt{w_i} - \sqrt{\hat{w}_l} \right)^2 + \left(\sqrt{h_i} - \sqrt{\hat{h}_l} \right)^2 \right] \\
&+ \sum_{i=0}^{S^2} \sum_{j=0}^{B} 1_{ij}^{obj} \left(C_i - \hat{C}_l \right)^2 + \lambda_{noobj} \sum_{i=0}^{S^2} \sum_{j=0}^{B} 1_{ij}^{nobj} \left(C_i - \hat{C}_l \right)^2 \\
&+ \sum_{i=0}^{S^2} 1_{ij}^{obj} \sum_{C \in class} \left(P_i(c) - P_l(\hat{c}) \right)^2 + \frac{\lambda}{2n} \sum_{w} w^2.
\end{aligned} \tag{7}
$$

The last term is the L2 regularization term, which is the sum of the squares of all the parameters w, divided by the sample size n of the training set. λ is the coefficient of the regular term, which weighs the proportion of the regular term and the other terms. There is also a coefficient 1/2, 1/2 is often seen, mainly for the

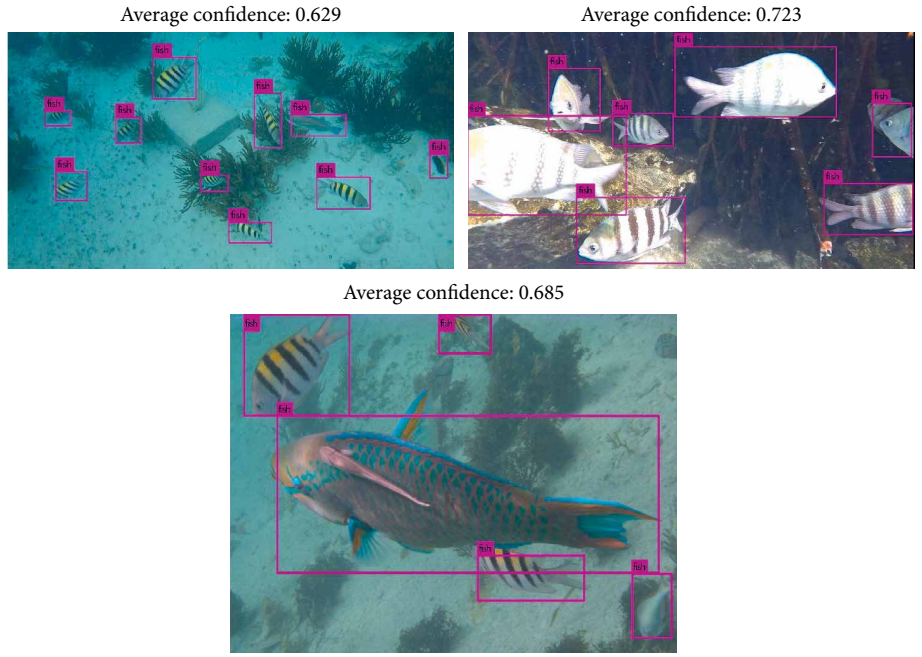


Figure 17: Enhancement of identification accuracy with dropout.

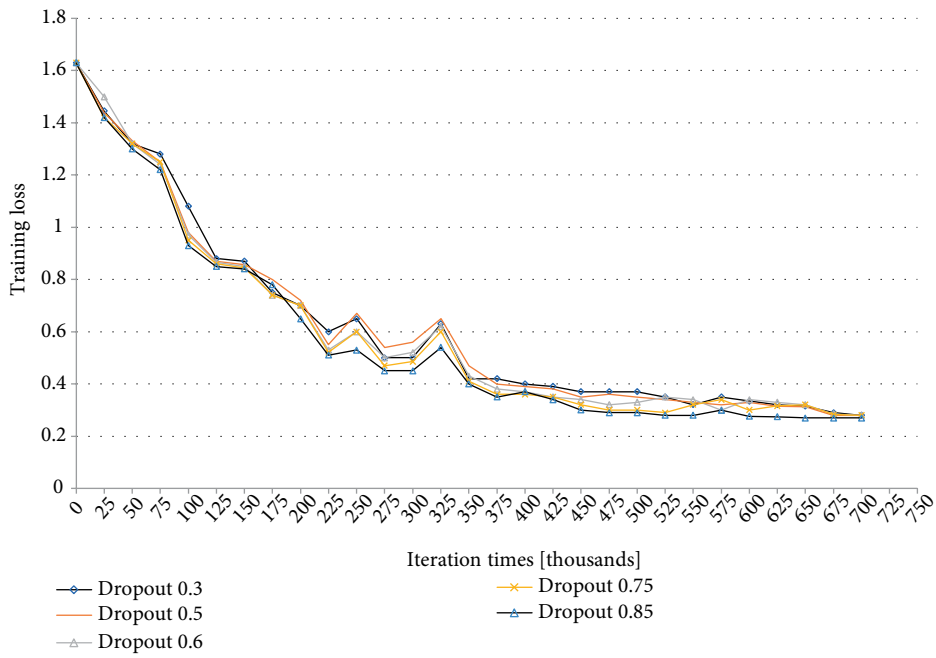


Figure 18: Effect of dropout on the training loss.

convenience of the results of the latter, the latter will produce a 2, multiplied by 1/2 just rounded up. The principle and procedure of how L2 reduce overfitting is in the reference [31].

5. Results and Discussion

The experiment implementations are based on the publicly available Tensorflow toolbox and Python language programming. The hardware platform is built on a GeForce GTX 745 GPU with 4G memory. The experiment is carried out by the three criterions aforementioned, the performance, training loss, and testing loss are compared respectively.

5.1. Experimental Results from Data Augmentation. The number of images is doubled by taking data augmentation transformation approach, which means this methodology could assist the machine to learn the feature more accurately. We used the test dataset to evaluate our model. It turns out that with data augmentation, the machine can identify the objects of interest

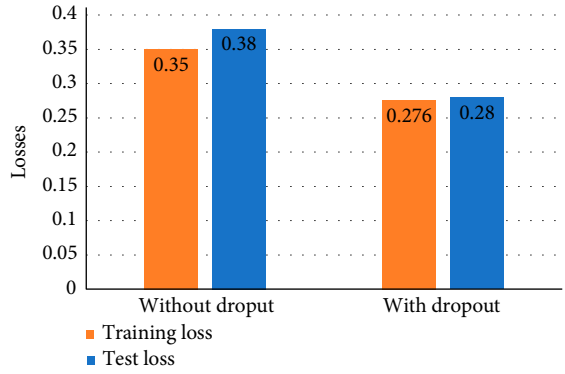


Figure 19: Effect of dropout on overfitting.

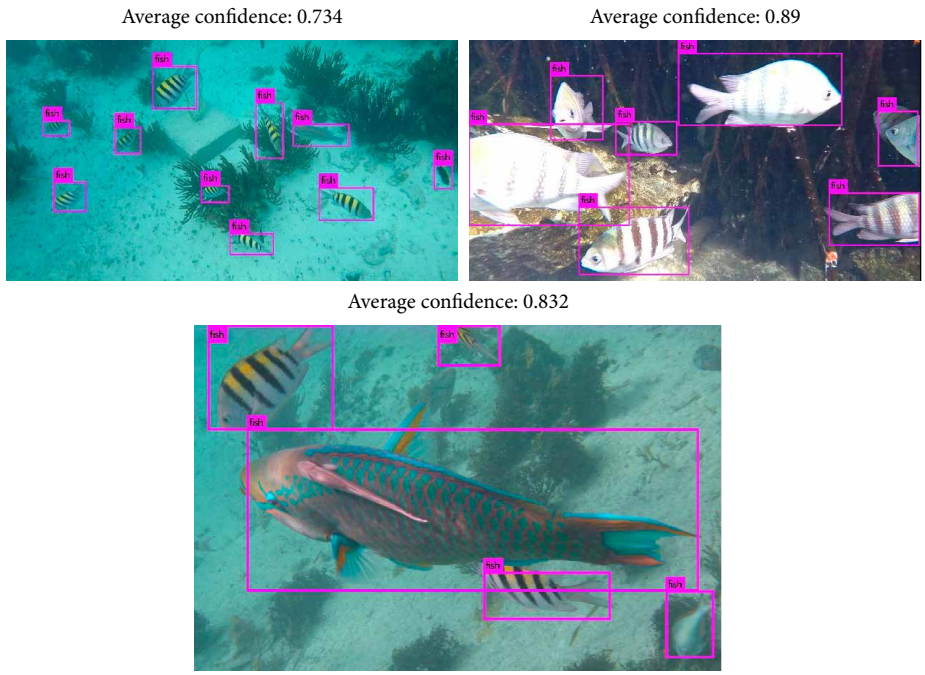


Figure 20: Enhancement of identification accuracy with the refinement of loss function.

more accurately than the result without data augmentation, the experiment result is clearly shown in Figure 14.

From Figure 14, it is clear that the fish in the image is identified accurately; the average confidence is 0.568, 0.65, and 0.59 respectively for the three sample images.

From Figure 15, it is observed that the final training loss of the proposed neural network model using original data is 0.35 while the final training loss using data augmentation is 0.46. This clearly demonstrates that the data augmentation transformation is much helpful to reduce the training loss.

Figure 16 illustrates the training loss with the increase on the number of iterations. The iteration times are set from 0 to 600. The difference between training loss and test loss is decreased from 1.6 to 0.46 and 0.35 respectively. The overfitting problem was solved to a great extent.

5.2. Experimental Results from Dropout. In this test, neurons in the hidden layers were randomly selected to be removed from this network. In this wary, a simplified deep neural network is obtained. Figure 17 shows the average confidence for each sample image is greatly improved; the effectiveness of identification by dropout approach is highly enhanced.

The training loss of dropout is illustrated in Figure 18. As we can see, the final training loss is 0.28 with dropout, and 0.35 without. The dropout algorithm can help reduce the training loss. In this specific application, when dropout is 0.85, the training loss is smallest all the time (Figure 18).

After conducted dropout approach, we can see from Figure 19, the difference between training loss and test loss decreased from 0.03 to 0.004. There is only a slight difference between training loss and test loss with dropout. The overfitting problem got resolved to a great extent. Therefore, the model we built is applicable.

5.3. Experimental Results from Loss Function Om Algorithm Performance. Authors used the CNN model with image

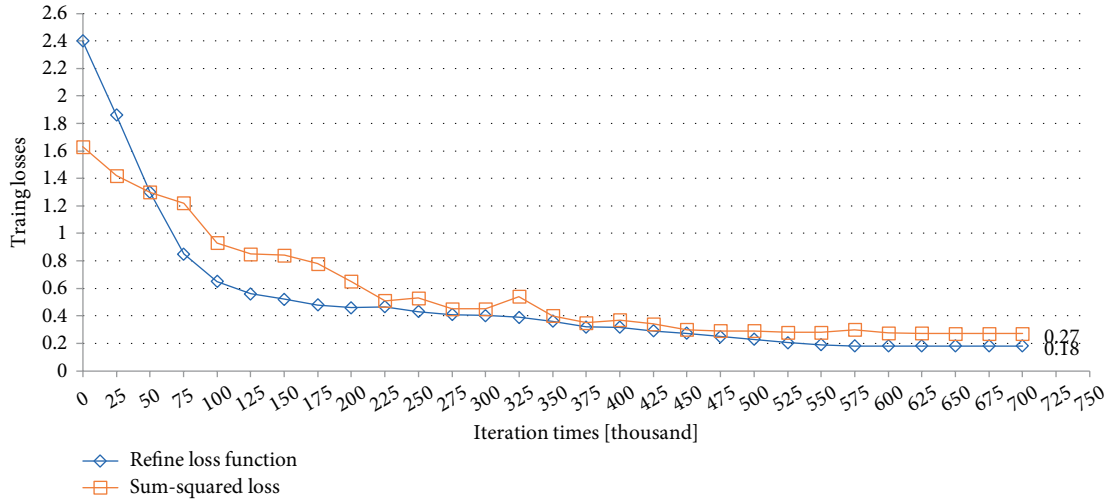


Figure 21: Effect of loss function on training loss.

segmentation and back-propagated the gradient of refined loss function and update the parameters in the network. With the refinement of loss function, the prediction get more accurate as clearly shown in Figure 20.

As shown in Figure 21, when the iteration times are 575, it converges. The final training loss is 0.18. However, with sum-squared loss function, the convergent point is 0.27 when the iteration times are 650.

5.4. Discussions. With the design and the choices of optimization, a deep learning based fish detection module was designed and simulated. With the improved accuracy and reduced processing time, it is very promising to adopt the proposed method to an AUV for implementation. The Tensorflow toolbox and Python programming interface are compatible with current advanced microcontroller platforms.

6. Conclusion

In this paper, authors built a neural network model to accomplish fish detection. To support the training process with enough dataset, the data augmentation approach was conducted. Dropout algorithm was selected to solve the overfitting problem. Moreover, loss function was refined to update the parameters inside the network. By these approaches, both the training time and the training loss were reduced dramatically. To summarize the contribution of this article: (1) Establish the data set to include real blur ocean water condition; (2) Revise loss function and other parameters in CNN to explore an applicable solution for fish detection; (3) The system is targeted at an embedded system for AUV design with all possible optimizations.

Data Availability

The program and image data used to support the findings of this study are available from the corresponding author upon request.

Conflicts of Interest

The authors declare that they have no conflicts of interest regarding the publication of this paper.

Acknowledgments

This work was sponsored by the United States NSF grants #1332566, #1827243, and #1411260.

References

[1] R. B. Wynn, V. A. I. Huvenne, T. P. Le Bas et al., "Autonomous underwater vehicles (AUVs): their past, present and future contributions to the advancement of marine geoscience," *Marine Geology*, vol. 352, pp. 451–468, 2014.

[2] M. Dinc and C. Hajiyev, "Integration of navigation systems for autonomous underwater vehicles," *Journal of Marine Engineering & Technology*, vol. 14, no. 1, pp. 32–43, 2015.

[3] Y. Zhou, S. Cui, Y. Wang, and C. Ai, "Design of autonomous underwater vehicle (AUV) control unit," in *2015 ASEE Gulf-Southwest Annual Conference*, pp. 25–27, ASEE Gulf-South, San Antonio, TX, 2015.

[4] Y. Zhou, S. Cui, Y. Wang, and L. Zhai, "A refined attitude algorithm for AUV based on IMU," in *15th International Conference on Scientific Computing (CSC'17)*, pp. 16–22, CSREA Press ©, Las Vegas, NV, 2017.

[5] SAUVC, https://sauvc.org/#competition.

[6] A. Cadena, P. Teran, G. Reyes, J. Lino, V. Yaselga, and S. Vera, "Development of a hybrid autonomous underwater vehicle for benthis monitoring," in *Proceedings of 2018 4th International Conference on Control, Automation and Robotics (ICCAR)*, pp. 20–23, IEEE, Auckland, New Zealand, 2018.

[7] M. Eichhorn, H. C. Woithe, and U. Kremer, "Parallel of path planning algorithms for auvs concepts, opportunities, and program – technical implementation," in *2012 Oceans - Yeosu*, MTS/IEEE, Yeosu, South Korea, 2012.

[8] H. Taka, T. Sasaki, and M. Wada, "Supporting system for fishry resource management utilizing convolutional neural

network," in *20th International Symposium on Woreless Personal Multimedia Communications (WPMC)*, pp. 442–447, IEEE, Bali, Indonesia, 2017.

[9] J. Kim, H. Cho, J. Pyo, and B Yu S-C Kim, "The convolution neural network based agent vehicle detection using forward-looking sonar image," in *OCEANS 2016 MTS/IEEE Monterey*, IEEE, CA, USA, 2016.

[10] F. Xu, X. Ding, J. Peng et al., "Real-time detecting method of marine small object with underwater robot vision," in *2018 OCEANS – MTS/IEEE Kobe Techno-Oceans (OTO)*, IEEE, Kobe, Japan, 2018.

[11] C.-F. Chien, Y.-J. Chen, Y.-T. Han et al., "AI and big data analytics for wafer fab energy saving and chiller optimization to empower intelligent manufacturing," in *Proceedings of 2018 e-Manufacturing & Design Collaboration Symposium (eMDC)*, pp. 1–4, IEEE, Hsinchu, Taiwan, 2018.

[12] J. Jia, "A machine vision application for industrial assembly inspection," in *Proceedings of 2009 Second International Conference on Machine Vision*, pp. 172–176, IEEE, Dubai, UAE, 2009.

[13] S. Biswas, Y. Wang, and S. Cui, "Surgically altered face detection using log-gabor wavelet," in *Proceedings of the 12th International Computer Conference on Wavelet Active Media Technology and Information Processing (ICCWAMTIP)*, pp. 154–157, IEEE, Chengdu, China, 2015.

[14] S. Cui, O. Ekwonnah, and Y. Wang, "The analysis of emotions over keystroke dynamics," in *2018 International Conference on Information, Electronic and Communication Engineering (IECE2018)*, pp. 28–29, DEStech Publications, Beijing, China, 2018.

[15] S. Cui, Y. Wang, and O. Ekwonnah, "Keystroke dynamics on user authentication," in *4th International Conference on Cybernetics (CYBCONF)*, pp. 5–7, Beijing, China, 2019.

[16] Eman Abdel-Maksoud, Mohammed Elmogy, and Rashid Al-Awadi, "Brain tumor segmentation based on a hybrid clustering technique," *Egyptian Informatics Journal*, vol. 16, no. 1, pp. 71–81, Cairo University, 2015.

[17] Y. Wang, Y. Lan, Y. Zheng, K. Lee, S. Cui, and J. Lian, "A UGV-based laser scanner system for measuring tree geometric characteristics," vol. 8905, in *Proceedings of 2013 SPIE International Symposium on Photoelectronic Detection and Imaging*, SPIE, Beijing China, 2013.

[18] B. Marr, "Key milestones of Waymo – Google's self-driving cars," https://www.forbes.com/sites/bernardmarr/2018/09/21/key-milestones-of-waymo-googles-self-driving-cars/#3831b2965369.

[19] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.

[20] BBC News, "Artificial intelligence: Google's AlphaGo beats Go master Lee Se-dol," 2016.

[21] About ImageNet, http://image-net.org/about-overview.

[22] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," *Advances in Neural Information Processing Systems*, vol. 25, pp. 1106–1114, 2012.

[23] J. Deng, A. Berg, S. Satheesh, H. Su, A. Khosla, and F. Li, "ImageNet large scale visual recognition competition 2012 (ILSVRC2012)," http://www.imagenet.org/challenges/LSVRC/2012/.

[24] Z.-Q. Zhao, P. Zheng, S.-T. Xu, and X. Wu, "Object detection with deep learning: a review," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 30, no. 11, pp. 3212–3232, 2019.

[25] R. Girshick, "Fast R-CNN," in *2015 IEEE International Conference on Computer Vision (ICCV)*, pp. 1440–1448, IEEE, Santiago, Chile, 2015.

[26] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: unified, real-time object detection," in *Proceedings of 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 779–788, IEEE, Las Vegas, NV, USA, 2016.

[27] Q. Peng, W. Luo, G. Feng et al., "Pedestrian detection for transformer substation based on gaussian mixture model and YOLO," in *2016 8th International Conference on Intelligent Human-Machine Systems and Cybernetics (IHMSC)*, pp. 562–565, IEEE, Hangzhou, China, 2016.

[28] S. Hassairi, R. Ejbali, and M. Zaied, "A deep convolutional neural wavelet network to supervised Arabic letter image classification," in *2015 15th International Conference on Intelligent Systems Design and Applications (ISDA)*, pp. 207–212, IEEE, Marrakech, Morocco, 2015.

[29] D. Zhang, G. Kopanas, C. Desai, S. Chai, and M. Piacentino, "Unsupervised underwater fish detection fusing flow and objectiveness," in *2016 IEEE Winter Applications of Computer Vision Workshops (WACVW)*, pp. 1–7, IEEE, Lake Placid, NY, USA, 2016.

[30] Convolutional neural networks for recognition, https://en.wikipedia.org/wiki/Kernel_(image_processing).

[31] Kernel (image processing), https://ujjwalkarn.me/2016/08/11/intuitive-explanation-convnets/.

[32] Deep learning and machine learning, https://ireneli.eu/2016/02/03/deep-learning-05-talk-about-convolutional-neural-network.

[33] M. Hu, Y. Yang, F. Shen, L. Zhang, H. Shen, and X. Li, "Robust web image annotation via exploring multi-facet and structural knowledge," *IEEE Transactions on Image Processing*, vol. 26, no. 10, pp. 4871–4884, 2017.

[34] J. Gaya, L. T. Gonçalves, A. Duarte, B. Zanchetta, P. Drews, and S. Botelho, "Vision-based obstacle avoidance using deep learning," in *2016 XIII Latin American Robotics Symposium and IV Brazilian Robotics Symposium (LARS/SBR)*, pp. 7–12, IEEE, Recife, Brazil, 2016.

[35] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: a simple way to prevent neural networks from overfitting," *Journal of Machine Learning Research*, vol. 15, pp. 1929–1958, 2014.