Underwater real-time target detection based on key frame and model compression

Wei Guo¹, Youbo Zhang¹,², Yue Zhou², Gaofei Xu¹ and Guangwei Li¹

¹ Institute of Deep-Sea Science and Engineering, Chinese Academy of Sciences, Hainan, 572000, China
² College of Engineering Science and Technology, Shanghai Ocean University, Shanghai, 201306, China
*Corresponding author’s e-mail: zhangyb@idsse.ac.cn (ORCID: 0000-0002-5894-5053)

Abstract. In order to improve the speed and keep high accuracy of UUVs object detection on embedded device. This paper proposes an acceleration strategy based on key frame extraction and model compression. Firstly, selecting shipwreck as target and creating a real-world underwater image dataset of the shipwreck. And this work applies both online and offline data augmentation to accelerate the model convergence and to improve the model generalization. Then, this paper builds a fusion algorithm with structural similarity, color histogram and image entropy to select the key frame of the captured video for removing the redundant information, and analyzing the reason for the complexity of the neural network and the method of model compression. Finally, this work cuts off unimportant structure of network based on channel and layer pruning to reduce model complexity and keeps the model with high accuracy by fine-tuning. The experiment shows the Billion Floating Operations of the compressed YOLOV4 model is reduced to 15.778. On the Nvidia Jetson TX2 embedded image processor, the average detection speed for video with 608×608 resolution can reach 15.12FPS. And the detection speed of the video composed of key frames is 2.98 times faster than the raw video.

1. Introduction
The ocean is an important part of human civilization, and more than half of international freight is transported through the ocean. A ship is a microcosm of the socio-economic development at that time for ocean trade often taking a long time. The life and belongings of the crew can highly reflect the social situation at that time. However, due to the limitations of navigation technology and extreme weather, shipwrecks have occurred from time to time. There are a large number of ancient shipwrecks scattered in the South China Sea alone. Finding and salvaging the shipwrecks can not only restore the social form at the time, but also enable better ocean exploration.

Unmanned underwater vehicles(UUVs) are widely used to detect close-range objects in the deep sea due to flexible and intelligent operation. Compared with other methods, UUVs with visual observation has unique advantages[1]. On the one hand, online target detection and tracking can be realized by real-time images. On the other hand, the photographed deep-sea video by UUVs can be online stored and offline analyzed to provide solid material for further research.

However, compared with ground target detection, underwater target detection has more challenges, with the following four limitations: 1) The underwater lighting environment is complex, and the
selective absorption of light by seawater will cause the color of objects in the underwater scene with varying degrees of change. 2) The underwater dataset is less. The lack of datasets will result in limited training samples. It is difficult to obtain a convergent detection model with small sample datasets. 3) The UUVs move slowly in the water, the collection video images have a high degree of repetition between frames, and repetitive detection will increase the resource and memory consumption of UUVs. 4) UUVs are limited by size and power consumption and can only carry embedded systems. Compared with high-power computing equipment in the laboratory, the computing power of embedded devices is relatively limited, and rapid recognition is the core of real-time target detection for UUVs[2]. Real-time target detection algorithms based on deep learning have made great progress and applications in real life. They are mainly divided into two categories: One is the two-stage that divides object detection into region proposals generation and bounding box regression. The other one is an end-to-end one-stage that directly predict bounding box[3]. Classical two-stage target algorithms are based on R-CNN series, including Fast-RCNN, Faster-RCNN, R-FCN and Mask RCNN etc., which have high detection accuracy but slow detection speed[4]. Typical representatives of one-stage algorithms include YOLO (You Only Look Once) series, SSD (Single Shot MultiBox Detector), RetinaNet, etc. One-stage algorithms directly detect objects through convolutional neural networks, and its detection speed is fast. But, it often requires a backbone network with stronger feature extraction capabilities to improve the detection accuracy[5-9]. In order to further improve the detection accuracy, CornerNet and CenterNet remove the anchor, and use HourglassNet to enhance the input feature. ResNet uses the "jump connection" method to deepen the feature extraction network[10], DenseNet widens the feature extraction network through dense connection[11], DPN(Dual Path Networks) absorbs the advantages of ResNet and DenseNet, and widen the network while deepening the network. However, as the depth and width of the network increase, the model parameters are gradually increasing, and the computing power of the hardware equipment carried by the UUV cannot meet the requirements of network inference.

In order to meet the needs of UUVs underwater real-time target detection with high accuracy, this paper chooses shipwreck as the detection target. Firstly, we build a benchmark dataset with real shipwreck images. The images in the dataset from the Hawaii Undersea Research Laboratory Archive(HURLA)[14], the collection videos of the NATURE FOOTAGE[15] and SQUID underwater dataset[16]. Then, this paper applies both online and offline data augmentation to accelerate the model convergence and obtain better detection model. And we fuse structural similarity, color histogram and image entropy to select the key frame of the captured video for removing the redundant information. Finally, this work prunes unimportant channels and layers of YOLOv4 model to reduce model complexity and keep model with high accuracy. The experimental results show that the complexity of the compressed YOLOv4 model is reduced to 15.778 BFLOPS(Billion Floating Operations), and the average detection speed on the Nvidia Jetson TX2 embedded image processor can reach 15.12FPS. The rest of this paper is organized as follows: In Section II, we establish a real-world shipwreck dataset and propose a fusion algorithm to extract online key frame of video. In Section III, we propose a novel method to compress model from channel and layer of the network. In Section IV, we evaluate our model from the quantitative and complexity metrics on embedded devices. And the summary of this paper in Section V.

2. Data pre-processing

2.1. Establishment of the target dataset
The deep learning target detection algorithm based on convolutional neural network has a strong dependence on data. The richer the training sample contains and the larger the number of samples, the better the generalization of the trained model. To avoid the influence of the image on the algorithm detection result, this paper builds a benchmark dataset with real underwater images. The images in the data set from the Hawaii Undersea Research Laboratory Archive(HURLA), SQUID and the collection
videos of the NATURE FOOTAGE. HURLA is an open source underwater image dataset with a large number of underwater species and marine heritage. The image is captured by UUVs with the single artificial light source in major seas around the world. And each image with detailed description and is divided into different categories according to their attribute in the HURLA. This paper selects 1780 shipwrecks images from the HURLA and attached website. The dataset of the SQUID includes abundant raw underwater images containing 57 stereo pairs from four different sites in Israel, two in the Red Sea and two in the Mediterranean Sea. This work selects 380 shipwrecks images from the SQUID. The dataset of NATURE FOOTAGE is collection videos of an underwater scene, this work downloaded videos related to the shipwreck and got 3642 images from the shipwrecks videos. In this paper, 5802 shipwreck images are collected through the above three basic datasets. And 90% of the data is divided into the training set and the remaining data is divided into the test set.

Figure 1. Example images from our dataset.

To further accelerate the convergence speed of model training and address the problem of dataset bias, this paper applies both online and offline data augmentation to prevent overfitting and to improve the model generalization. Our offline data augmentation consists of four transformations: zooming (50%N-55%N, N is the number of images in the training set), vertical/horizontal translations (50%N-55%N), rotation (50%N-55%N), shearing (50%N-55%N). With these operations, we increase the training set 3-4 folds.

Figure 2. The transformations of the original data used in offline data augmentation.

And our online data augmentation uses mosaic method to increase the richness of the training sample and let limited data samples maximize the value of model training. Mosaic data enhancement reads four images at a time, randomly converts the original image, and then stitches the four images into a complete image in random order and sends it to the detection model. Compared with other online data enhancement methods, the Mosaic enhancement method makes the image input into the network at a
time from the original single image to 4 images, which effectively enriches the number of target samples in the training stage[5].

2.2. Online video key frame extraction
UUVs move often slowly in the water. In the continuously captured video, there will be a high similarity between adjacent frames of video. Each frame of image is enhanced and detected, it will increase hardware computing load and system resource consumption. Video is composed of frames of image. The color and texture information of frames of image is different, and the image content is also different[17].

In order to accurately extract key frames, this paper uses a fusion algorithm based on the structural similarity(SSIM), color histogram(CH) and image entropy(ENTR) to extract online key frame of the collected video.

SSIM uses the brightness, contrast, and structural information of the video frame to indicate the degree of similarity between the two frames. The larger the value, the higher the similarity. The similarity of SSIM is expressed as:

\[ S_{SSIM} = \frac{(2\mu_u\mu_v + C_1)(2\sigma_{uv} + C_2)}{\mu^2_u + \sigma^2_u + \mu^2_v + \sigma^2_v}, \tag{1} \]

Where \( u \) is the \( u \)-th frame of the captured video \( V \), \( v \) is the \( v \)-th frame of the captured video \( V \). \( \mu_u, \mu_v \) respectively are the mean value of the \( u \)-th and \( v \)-th frame image, \( \sigma_u, \sigma_v \) respectively are the variances of \( u \) and \( v \), \( \sigma_{uv} \) is the correlation coefficient of \( u \) and \( v \). \( C_1, C_2 \) are constants to prevent the denominator from being 0. And the range of \( S_{SSIM} \) is from 0 to 1.

At present, most underwater cameras are RGB mode that the generated image with RGB three color channels. In order to better compare the images between frames based on CH, we convert the video frames to HSV color space. In the HSV color space, H means hue, S means saturation, and V means brightness. The range of H is \([0^\circ, 360^\circ] \), where \( 0^\circ \) means red, \( 120^\circ \) means green, \( 240^\circ \) means blue. The S represents the depth of the color, and the value range is \([0,1]\). It is generally believed that the higher the S value, the darker the color. The brightness V indicates the degree of lightness and darkness of the color, and the range is also \([0,1]\). As the value of V increases, the color gradually becomes darker[18].

![RGB to HSV](image)

**Figure 3.** RGB to HSV.

As shown in Figure 3, the HSV color space has a stronger contrast than RGB. The color histograms values of the same image in the HSV color space are more outstanding and dispersive, which is good
for the comparison of similarity between images. Defining respectively $H_u(RGB2HSV(u))$ and $H_v(RGB2HSV(v))$ to be the color histograms of the $u$ and $v$, and the similarity of CH between the two frames can be expressed:

$$S_{CH} = \frac{H_u(RGB2HSV(u)) \cdot H_v(RGB2HSV(v))}{H_u(RGB2HSV(u)) \times H_v(RGB2HSV(v))}$$  \hspace{1cm} (2)$$

Where $RGB2HSV(V)$ is the function that converting the i-th frame image in the captured video from RGB to HSV color space. $S_{CH}$ represents the cosine distance between 0 and 1, the smaller the value, the more similar the two frames of images.

Entropy is used to measure the uncertainty of information. The more information an object contains, the higher its entropy value. Therefore, we can use the image entropy to means the richness of the content contained in an image[19]. The two frame images are very similar in content and texture, indicating that the information entropy of the two images is also very close. The entropy of an 8-bit gray-scale image is defined as follows:

$$ENTR = -\sum_{j=0}^{255} p_j \log_2 p_j$$  \hspace{1cm} (3)$$

where $p_j$ is the probability of the gray level $j$ in a pixel of the image, which is calculated by:

$$p_j = \frac{\sum_{i=1}^{M} (I(x) == j)}{\sum_{k=0}^{255} \sum_{i=1}^{M} (I(x) == k)}$$  \hspace{1cm} (4)$$

where $I(x)$ represents the pixel value of the i-th frame image and $M$ is the total number of pixels in the image. To measure the image entropy of an RGB image, this work separately calculates the entropies of the three color channels, then calculate the average of them as the image entropy of the whole image. And the similarity of ENTR between the two frames can be expressed:

$$S_{ENTR} = \left| \frac{1}{c} \sum (ENTR(u^c)) - \frac{1}{3} \sum (ENTR(v^c)) \right|, c \in (R,G,B)$$  \hspace{1cm} (5)$$

And the smaller the $S_{ENTR}$ value, the more similar the two frames of images.

Fusion methods are often divided into splicing, weighted, coefficient feature fusion. Due to the color space difference between SSIM, ENTR and CH, this paper chooses the weighted fusion method to compare the similarity between the two frame images. The final online video key frame extraction algorithm is as follows:

$$S_f = -\sigma S_{SSIM} + \phi S_{CH} + \psi S_{ENTR}$$  \hspace{1cm} (6)$$

where $\sigma, \phi$ and $\psi$ are weighed factors, this paper respectively sets $\sigma, \phi$ and $\psi$ as 1, 1 and 5 based on experimental results. And the smaller the value, the more similar the two frames of images.
Figure 4. Example of key frame verification

The overall key frame extraction algorithm is as follows:

**Algorithm: Key frame extraction**

**Input:** $V, \xi = 0.9, \omega = 1, vi = 1, V_0 = K$

**Output:** $G$

$G_0 = V_0$

for iteration from 1 to $\Psi$

if $S_f(V_i, K) \in [\xi, \omega]$

$K = V_i$

else:

$G_i = V_i$  

$vi++$

end for

return $G = \{G_0, G_1, \cdots, G_\Psi\}$

In the algorithm, $\Psi$ is the total frame number of raw video, $V_0$ is the first frame of the $V$, $K$ is the reference frame, and $G$ is the set of key frame image.

3. **Model compression**

3.1. **Neural network compression survey**

Convolution Neural Networks (CNNs) are widely used in computer vision tasks like object detection, classification, and segmentation. CNNs are designated as “Deep” because they are made of many layers with a large spread of connections between layers[12]. The deep learning target detection model is composed of neurons based on a specific network connection structure, and uses error back propagation to update the weight of each neuron. As the number of training iterations increases, some neurons in the model have less and less effects on the results, and have no contribution to target recognition, but the existence of the network structure will increase the model volume. The work of model compression is to eliminate unimportant neurons and their network connections in the model under the premise of ensuring accuracy. The main compression methods currently used for target detection tasks including model pruning, quantification, network architecture search, knowledge distillation and matrix factorization.
Table 1. Summary of neural network compression methods.

| Method              | Principle of Description                                      | Advantage                          | Disadvantage                                      |
|---------------------|--------------------------------------------------------------|------------------------------------|---------------------------------------------------|
| Pruning             | Cut off unimportant parameters and connections               | Flexible operation, low            | Fine-tune, training time increases                 |
|                     |                                                              | precision loss                     | The accuracy of the model has dropped severely and the accuracy is difficult to recover |
| Quantization        | Reduce parameter bit width                                   | Reduced model calculations         | It is difficult to set search indicators, the search process takes a long time, and training costs are high |
| Network Search      | Automatically search out a network model that meets the      | High model accuracy                | Knowledge transfer standards are difficult to determine and only apply to classification |
|                     | parameters and accuracy requirements                         |                                    | Training is difficult and costly                  |
| Knowledge           | Transfer knowledge from complex models to easy models       | A few parameters are used for the calculation |                                    |
| distillation        |                                                              | High model accuracy                |                                    |
| Matrix Factorization| Decreases the size of CNNs filters                           |                                    |                                    |

3.2. Selection of baseline

Neural network compression firstly needs to select a basic model. In order to make the compressed model ensure the detection speed while maximizing the detection accuracy, this paper chooses YOLOV4 as the basic model. YOLOV4 is currently one of the best target detection in one-stage algorithms, consisting of 162 convolutional layers. The backbone network uses the CSPDarknet53 structure and mish activation function for feature extraction, and the feature prediction part uses the LeakyRelu activation function. Experiments have shown that different activation functions in the feature extraction and prediction stages can effectively improve the accuracy of target detection[5]. In order to further improve the accuracy of target recognition, YOLOV4 uses the PANet to replace the FPN structure in V3. At the same time, the multi-scale feature size of YOLOV4 has changed to 76 × 76, 38 × 38, 19 × 19.

YOLOV4 is an anchor-based target detection algorithm that divides the input image into a $S \times S$ grids. Some anchor boxes are empirically set on each grid, and the final bounding box(bbox) regression is that predicting the distance between the anchor box and the ground truth box(GT). However, the original anchor of YOLOV4 is obtained based on the Pascal VOC dataset. The VOC2012 data set includes 20 category images. The training set has a size of 11540 and contains 27450 object instances. The size of the objects varies greatly. The defaulted anchor box selected by YOLOV4 is ([12 16], [19 36], [40 28], [36 75], [76 55], [72 146], [142 110], [192 243], [459 401]), directly using the defaulted anchor for underwater target detection will affect the model convergence and detection accuracy. In order to ensure the accuracy of the compressed model, it is necessary to improve the detection accuracy of the baseline as much as possible. This paper uses the K-means++ algorithm to conduct cluster analysis on the ground truth box with annotation information to determine the new anchors belonging to the target dataset.

$$\text{dis}(\text{bbox}, \text{cent}) = 1 - IoU(\text{bbox}, \text{cent}), IoU = \frac{\text{bbox}_{\text{true}} \cap \text{bbox}_{\text{true}}}{\text{bbox}_{\text{true}} \cup \text{bbox}_{\text{true}}}$$  \hspace{1cm} (7)

Where $IoU$ is the intersection over union of GT and prediction box, $\text{dis}$ represents the category distance, and $\text{cent}$ is the bbox calculated by the K-means++ algorithm. The cluster analysis of GT in
the data set can be realized based on equation (7). In order to more accurately determine the number of anchors of the target dataset, the result of the cluster analysis is calculated by equation (8).

\[
\text{Acc} = \arg \max_{i=1}^{W} \text{IoU}(\text{box}, \text{cent})
\]  

(8)

Where \( W \) is the value of \( K \), \( K \) is the number of clusters. According to the clustering analysis results in Table 2, when the \( K \) is 8, the clustering accuracy is 81\%, and the clustering accuracy does not change much as the \( K \) increases. Therefore, this paper chooses the number of anchors as 9, and the anchor box is redefined as ([52 51], [92 106], [164 146], [162 255], [306 183], [223 315], [354 269], [290 360], [378 375]).

| \( K \) | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
|-------|---|---|---|---|---|---|---|---|---|----|
| \( \text{Acc} \) | 0.45 | 0.62 | 0.69 | 0.71 | 0.72 | 0.75 | 0.78 | 0.80 | 0.81 | 0.81 |

### 3.3. Model compression algorithm design

In order to ensure the accuracy of compression model detection, this paper selects the pruning method to compress the basic model based on the basic network structure and energy consumption of UUVs. BN (Batch Normalization) is an important part of convolutional neural networks. It normalizes the convolution output to ensure that the input and output of each convolution layer are in the same distribution, thereby speeding up network convergence.

\[
\gamma^{(\eta)} = \gamma^{(\eta)} \times \frac{x^{(\eta)} - \bar{x}^{(\eta)}}{\sqrt{(\delta^{(\eta)})^2 + \epsilon}} + \beta^{(\eta)},
\]

where \( \eta \) is a channel of the convolution layer, \( \gamma^{(\eta)} \), \( \beta^{(\eta)} \) is the corresponding scaling and Offset factor. The BN structure is located after the convolutional layer in the network structure, and the scaling factor on the BN layer can be used to determine the importance of the channel and the convolutional layer without adding any parameters [13]. In order to obtain the best compression model, we propose a model pruning method based on channel and layer pruning. The compression process is divided into four steps: basic training, sparse training, model pruning and fine-tuning. The specific operations are as follows:

**Step1:** Basic training. Choosing YOLOV4 as baseline. \( \text{Loss}_{\text{raw}} \) is loss function. Using YOLOV4 for model training with target dataset. When \( 0 < \text{Loss}_{\text{base}}(\text{baseline}) \leq 1 \) and \( \text{Loss}_{\text{base}} \) stable change, we stop basic training.

**Step2:** Select the scaling factor on the BN layer to measure the importance of the channel, and for the \( L1 \) regularization with \( \gamma \), it can be known from equation (10) that the \( L1 \) space is composed of line to generate sparse values.

\[
L1(\gamma) = |\gamma^{(1)}| + \cdots + |\gamma^{(\eta)}| + \cdots + |\gamma^{(ch)}|
\]

(10)

Where \( \gamma^{(\eta)} \) is \( \eta \)-th channel and \( ch \) is the total number of channels of the \( i \)-th convolution layer.

**Step3:** Sparse training. \( L1 \) regularization is added to the \( \text{Loss}_{\text{base}} \). Constrain the update range of the model parameters to form a new loss function in the sparse training process of the model.

\[
L = \text{Loss}_{\text{base}}(\text{baseline}) + \alpha \sum g(\gamma),
\]

(11)

Where \( g(\gamma) \) is the \( L1 \) regularization of the scaling factor \( \gamma \), \( \alpha \) is the penalty coefficient used to control the degree of regularization.
Step4: Channel pruning. After sparse training, some channels in the network will tend to 0. Sorting $\gamma$ of each channel in the convolutional layer and defining the initial channel pruning threshold as $\tau_{ch}$:

$$
\begin{align*}
\tau_{ch} &= \gamma'[\text{floor}(0.8ch)] & \gamma'[\text{floor}(0.8ch)] < \tau_{ch} \\
\tau_{ch} &= \gamma'[\text{floor}(0.05ch)] & \gamma'[\text{floor}(0.05ch)] \geq \tau_{ch}
\end{align*}
$$

Where $\text{floor()}$ is the round down operation. And we cut off the channel and its connection when $\gamma^{(ij)} \leq \tau_{ch}$.

Step5: Layers pruning. Calculating the mean value of $\gamma$ in each convolutional layer, it is defined as

$$
\bar{\gamma} = \frac{1}{ch} \sum_{j=0}^{ch} \gamma^{ij}.
$$

$$
\begin{cases}
\bar{\gamma} = 0 & \min_{i \in L} \{ \bar{\gamma} \} + \delta < \tau_{lay} \\
\end{cases}
$$

Where $E$ is the number of layer, $\delta$ is the threshold adjustment factor. The initial layer pruning threshold as $\tau_{lay}$. And we cut off the corresponding layer when $\gamma^{(ij)} = 0$.

Step6: Model fine-tuning. Reset the hyperparameters of the compressed model obtained after channel and layer pruning. Training and fine-tuning the compressed model with the original data set to get the final compression model that Comp.YOLOV4.

Due to the different training results of each layer of the network, using the different pruning rate in the channel pruning stage can effectively avoid “over-cutting” and “less-cutting”. At the same time, the importance of each convolutional layer in the model after channel pruning is sorted, and the unimportant layers and their connections are pruned again by layer pruning. Compared with other methods, our pruning strategy in this paper can more effectively identify the important part of the network, considering both the ratio of the layer and channel and its importance, and further improve the model compression rate.

4. Experiment and analysis

4.1. Evaluation of baseline

In order to further analyze the performance of the baseline, this paper selects precision($P$), recall($R$), Intersection over Union($IoU$), F1-score, mean average precision($mAP$), and volume of model($Vol$) to evaluate baseline. YOLOV3, YOLOV3-SPP, YOLOV3-tiny and YOLOV4 are selected for the basic training of the model. Due to the small number of target dataset, it is difficult to obtain generalized results by common training. This paper adopts transfer learning strategy and uses the weights trained on the ImageNet dataset as the initial weights of the model. We define the input image resolution with 608×608, the batch size with 16, the momentum with 0.949, the initial learning rate with 0.001, the weight decay with 0.0005, and the number of iterations with 10000. The training platform is configured with 64G Intel i7-9700K processor and 8G Nvidia RTX2080 graphics card. The statistics of the basic training performance are shown in Table 3. The black bold items in the table are the best in a single category.

| Baseline | $P$ | $R$ | $IoU$ | $F_1$ | $mAP$ | Loss | $Vol$ |
|----------|-----|-----|-------|-------|-------|------|-------|
| YOLOV3   | 89% | 75% | 78.04%| 81%   | 76.43%| 0.6528| 246.3MB|
| YOLOV3-SPP| 92% | 60% | 81.98%| 72%   | 67.93%| 0.5436| 250.5MB|
| YOLOV3-tiny| 54% | 34% | 32.98%| 38%   | 30.78%| 1.4550| 34.7MB |
| YOLOV4   | 93% | 82% | 84.23%| 90%   | 82.51%| 0.4965| 256.0MB|

Table 3. Statistics of the basic training performance of models
YOLOV4 is much better than other models, and the overall performance of the model is the best based on Table 3. The accuracy of the compressed model depends on the accuracy of the model after basic training. Experiments show that choosing YOLOV4 as the basic model can better ensure that the compressed model has higher detection accuracy.

4.2. Channel and layer pruning
The Vol of YOLOV4 is relatively large from Table 3. In order to realize UUVs real-time online target detection, channel and layer pruning are adopted on the YOLOV4 model after basic training. We define the iterations of the sparse training is 400, the initial learning rate is 0.001, and the learning rate is reduced by 10 times when the number of iterations reaching 80% of the total iteration. And other parameters remain the same as the basic training. The statistical result of the fine-tuned model is shown in Table 4.

| Comp_YOLOV4   | $\alpha$ | $\tau_{ch}$ | $\tau_{lay}$ | $\delta$ | $F_1$ | mAP | Loss | Vol     |
|---------------|----------|-------------|--------------|----------|-------|-----|------|---------|
| YOLOV4-m-10000 | $10^{-4}$ | 0.10        | 0.5          | 0.05     | 71%   | 70.85% | 0.9650 | 18.5MB  |
| YOLOV4-pr-10000 | $10^{-4}$ | 0.10        | 0.5          | 0.05     | 72%   | 75.66% | 0.7450 | 18.5MB  |
| YOLOV4-m-10000 | $10^{-5}$ | 0.12        | 0.55         | 0.05     | 76%   | 73.77% | 0.7400 | 17.5MB  |
| YOLOV4-pr-10000 | $10^{-5}$ | 0.12        | 0.55         | 0.05     | 79%   | 80.90% | 0.5746 | 17.5MB  |
| YOLOV4-m-10000 | $10^{-4}$ / $10^{-5}$ | 0.08      | 0.45        | 0.05     | 83%   | 81.98% | 0.6133 | 15.5MB  |
| YOLOV4-pr-10000 | $10^{-4}$ / $10^{-5}$ | 0.08      | 0.45        | 0.05     | 84%   | 81.32% | 0.6518 | 12.6MB  |

In Table 4, model-m-10000 represents that using weights of the pruning model to fine-tune and 1000 is the iteration of fine-tuning. And the model-pr-10000 represents that using weights of the pre-trained model on ImageNet to fine-tune. $\alpha=10^{-4} / 10^{-5}$ means that the regularization penalty coefficient is $10^{-4}$ in the first 199 iterations of sparse training, and the penalty coefficient in the subsequent iterations is $10^{-5}$. From Table 4, the compression ratios of the model under three different pruning strategy are 13.83, 14.63 and 20.32. And we can get the best pruning model in model-pr-10000 when $\alpha=10^{-4} / 10^{-5}$, $\tau_{ch}=0.08$, and $\tau_{lay}=0.45$. In this mode, $F_1$ and Loss are basically the same as the basic model, and the mAP on the test set can reach 81.32%. The best model structure is shown in Figure 5.

**Figure 5.** Structure of Comp_YOLOV4
4.3. Result of embedded platform detection
The detection result calculated by the high-power and high-load platform is challenging to be applied in the UUVs, which are limited in space and power. Based on actual requirements, we choose Nvidia image processor Jeston TX2 as the experimental tool and use Python language to program. TX2 core board size is 50×87 mm, power consumption under typical load is only 7.5W, CPU uses ARM Cortex-A57, GPU uses Nvidia Pascal GPU, which can meet the power consumption and size requirement of UUVs.

Table 5. Detection performance of the TX2

| Model        | KFE | Size    | avg FPS | BFLOPS  | Time |
|--------------|-----|---------|---------|---------|------|
| YOLOV3       | N   | 608×608 | 2.8     | 139.512 | 562s |
| YOLOV3-SPP   | N   | 608×608 | 2.8     | 140.320 | 578s |
| YOLOV4       | N   | 608×608 | 2.9     | 127.248 | 510s |
| Comp_YOLOV4  | N   | 608×608 | 15.12   | 15.778  | 97.8s|
| Comp_YOLOV4  | Y   | 608×608 | 15.12   | 15.778  | 32.74s|

As shown in Table 5, KFE is the option of Key frame extraction, Size is the image resolution of the network input, avg FPS is the average detection speed and Time is the spend that detecting an underwater video of 45 seconds. The average speed of the uncompressed model on the embedded device is only 2.6-2.8FPS, and high complexity reduces the detection speed. The complexity of the compressed model is significantly reduced. The detection speed of the image with the same resolution on the TX2 is faster 5.8 times than uncompressed model, reaching 15.12FPS, which can basically realize online real-time target detection requirements of UUVs by key frame extraction and model compression. The detection result is shown in Figure 6.

Figure 6. Results of the target detection

5. Conclusion
To solve the problem that UUVs cannot achieve underwater real-time online target detection due to the limitation of space and power consumption. In this paper, we improve the speed of UUVs object detection on embedded device from key frame extraction and model compression. To ensure the accuracy of the compression model, we choose YOLOV4 as the baseline. And selecting shipwreck as the target and using data enhancement to improve the generalization of data. And we fuse structural similarity, color histogram and image entropy to select the key frame of the captured video for removing the redundant information. The key frames are only sent to the network for target detection, reducing repetitive detection and improving detection speed. To further speed up model detection, we compress YOLOV4 based on channel and layer pruning. And the pruning strategy of two-way can effectively avoid over-cutting and less-cutting.

By comparing the baseline with its compressed model, it can be concluded that the cost of the increase of model speed is reduced accuracy. Furthermore, it is to remove the convolutional layers and channels in the network. It is difficult to improve the accuracy of by fine-tuning when the model is pruned largely. Experiments show that the compressed Comp_YOLOV4 model can detect underwater object on embedded devices with high accuracy and fast detection speed.
Acknowledgments
This work was supported by the Natural Science Foundation of Hainan Province (Grant No. 2019RC260), the National Key Research and Development Project (Grant No. 2020YFC1521704), the Science and Technology Cooperation Project of Sanya City (Grant No. 2019YD01).

References
[1] Manley, J.E.(2016) Unmanned Maritime Vehicles, 20 years of commercial and technical evolution. OCEANS 2016 MTS/IEEE Monterey, Monterey, CA, pp. 1-6.
[2] Lin S., Zhao Y.(2020) Review on Key Technologies of Target Exploration in Underwater Optical Images[J]. Laser & Optoelectronics Progress, 57(6): 060002.
[3] Ren S., He K., Girshick R.(2017) Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks[J]. IEEE Transactions on Pattern Analysis and Machine Intelligence, 39(6): 1137-1149.
[4] He K M., Gkioxari G., Dollár P. Mask R-CNN. https://arxiv.org/abs/1703.06870.(2020-12-20).
[5] Bochkovskiy A., Wang C Y., Mark Liao H Y. YOLOv4: Optimal Speed and Accuracy of Object Detection. https://arxiv.org/abs/2004.10934v1.(2020-12-20).
[6] Redmon J., Farhadi A.(2017) YOLO9000: Better, Faster, Stronger. IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Honolulu, HI, pp.6517-6525.
[7] Redmon J., Farhadi A. Yolov3: An incremental improvement. https://arxiv.org/abs/1804.02767. (2020-12-20).
[8] Liu W., Anguelov D., Erhan D.(2016) SSD: Single shot multibox detector. Proceedings of the European Conference on Computer Vision (ECCV), pp. 21–37.
[9] Lin T Y., Goyal P., Girshick R.(2017) Focal loss for dense object detection. Proceedings of the IEEE International Conference on Computer Vision (ICCV), pp. 2980–2988.
[10] He K M., Zhang X Y., Ren S Q.(2016) Deep residual learning for image recognition. Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp.770–778.
[11] Huang G., Liu Z., Maaten L. Densely Connected Convolutional Networks. https://arxiv.org/abs/1608.06993.(2020-12-20).
[12] Goel A., Tung C., Lu Y H. A Survey of Methods for Low-Power Deep Learning and Computer Vision. https://arxiv.org/abs/2003.11066.(2020-12-20).
[13] Liu Z., Li J G., Shen Z Q.(2017) Learning efficient convolutional networks through network slimming. Proceedings of 2017 IEEE International Conference on Computer Vision. Venice, Italy. pp. 2755–2762.
[14] http://www.soest.hawaii.edu/HURL/HURLarchive/index.php.(2020-12-20).
[15] https://www.naturefootage.com/stock-video-footage?fis=shipwreck. (2020-12-20).
[16] Berman D., Levy D., Avidan S., and Treibitz T.(2020) Underwater Single Image Color Restoration Using Haze-Lines and a New Quantitative Dataset. IEEE Transactions on Pattern Analysis and Machine Intelligence. doi: 10.1109/TPAMI. p. 2977624.
[17] Cai S., Zhang Q., Wang Q., Lei Y., and Yang J.(2020) Multi-frame Dimensionality-Reduction Difference Method for Extracting Key Frames of Video. 2020 IEEE 44th Annual Computers, Software, and Applications Conference (COMPSAC), Madrid, Spain, pp. 1466-1470.
[18] Xu B., Qi R., and Jiang B.(2020)Adaptive Fault-Tolerant Control for HSV with Unknown Control Direction. IEEE Transactions on Aerospace and Electronic Systems, 55(6): 2743-2758.
[19] Li M., Ma K., You J., Zhang D., and Zuo W.(2020) Efficient and Effective Context-Based Convolutional Entropy Modeling for Image Compression. IEEE Transactions on Image Processing, 29:5900-5911.