FAST RECOGNITION ALGORITHM OF PAVEMENT CRACKS BASED ON DEEP CONVOLUTION AND UNSUPERVISED CLUSTERING

Ru YANG[1], Zhentao QIN[2]*, Weibo WANG [1], Remi DATRIGANT [3]

With the large-scale operation of highways in China, the daily maintenance and management of highways has restricted the road operation. If the pavement cracks can be repaired in time before being deteriorated into pits, the maintenance cost can be greatly saved. This work proposed a new fast recognition algorithm for pavement cracks based on unsupervised clustering. First, the normalized image of pavement cracks was sliced into two-scale image blocks using the sliding window algorithm. Then a deep convolutional self-encoder was constructed to extract and reconstruct the image data. Finally, the identification model with the encoder and the cluster layer detected the pavement cracks. The results showed that compared with the current mainstream algorithms, the proposed method has better recognition with faster speed.

Keywords: Pavement cracks; Deep convolution; Unsupervised Clustering; Self-encoder

1. Introduction

In the intelligent transportation system, the automatic detection of road damage has always been the focus and difficulty of research [1,2]. Faced with the growing demand for highway construction, manual inspection has the disadvantages of low operational efficiency, large subjective impact, and high risk, which cannot meet the requirements for fast detection of road damage. Although the traditional vision-based target localization and detection is simple and easy to use [3], it does not consider the change of image environment as well as the influence of illumination, noise and texture on crack recognition. Therefore, it is difficult to achieve a stable effect.

Although the system using deep learning can accomplish this task, deep learning requires a large amount of labeled data to train the model, which is a cumbersome task [4]. In recent years, Alexnet [5], GoogLeNet [6], Vgg [7] and other networks have achieved breakthroughs in solving classification, and Faster

1 School of Civil and Architecture Engineering, Panzhihua College, Panzhihua, China
2 School of Mathematics and Computer Science , Panzhihua College, Panzhihua, China
3 Technical University of Cluj-Napoca, Romania
* corresponding author: Zhentao Qin, e-mail:309507443@qq.com

R-CNN [8], YOLO [9], SSD [10] and other networks have achieved certain success in solving target detection. However, these methods are not ideal for the recognition of pavement cracks. For example, R-CNN cannot obtain specific crack position.

In order to solve this problem, Li liangfu [11] et al. proposed the image to be detected is divided into 16x16 image slices, and the classification model (Deep-bridge-crack-classify, DBCC) is used for each image slice to identify the background and foreground of crack image. The crack foreground is then mapped to the original image to obtain the location of the image of crack. Compared with traditional algorithms, this algorithm has better generalization and higher recognition rate; however, there is still a lot of data to mark.

The work used the sliding window algorithm to divide the normalized image of pavement cracks into two-scale image blocks. A new method based on unsupervised learning was proposed, and accuracy was ensured without labeling the image. At the same time, the method proposed in the work belongs to the unsupervised classification with faster recognition speed.

2. Methodology

A deep convolutional self-encoder was constructed to extract and reconstruct image data. Then, after the deep convolution of self-encoding encoder, the cluster layer was connected for the classification of the image (See Fig. 1). This method required normalization of the image, and the segmented image was sent to a deep learning model for classification. After completing the classification, the images classified as the foreground of cracks were remapped to the original image to obtain the coordinates of cracks.

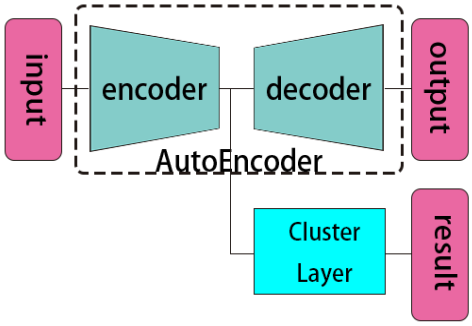

Fig. 1. The basic structure of the self-encoder

2.1 Construction of multi-scale sliding window

By analyzing a large number of images of pavement cracks, it was found that the cracks in the image occupy a small area of the original image. If the

image is segmented according to a single scale, a image of 1,024*1,024 needs to classify 4,096 image blocks with a resolution of 16*16, and most of the images are background images with no interest. It wastes computing resources, which seriously affect the performance of algorithm.

In order to solve this problem, the work used the window sliding of 64*64 and 16*16 to segment the image. The original image was segmented into 64*64 blocks, and the algorithm only needed to classify 256 image blocks. After that, 256 image blocks containing cracks were segmented and classified, with the new scale of 16*16 (See Fig. 2).

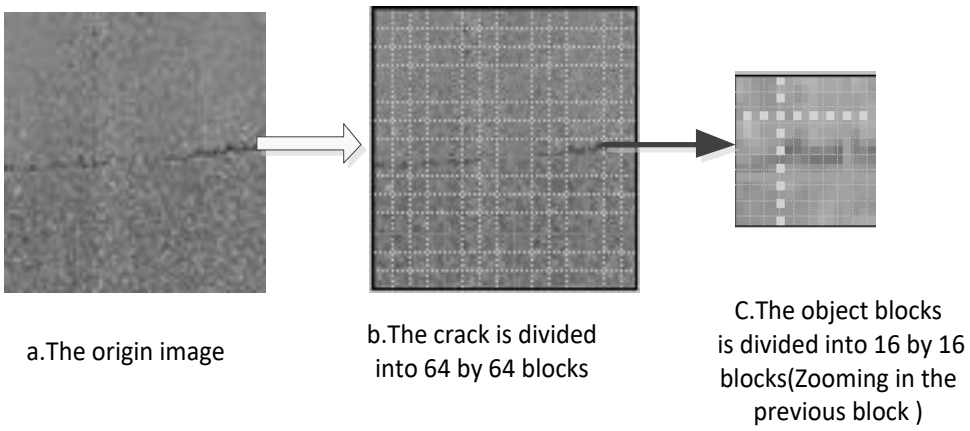

Fig. 2. Construction intention of multi-scale sliding window of pavement crack surface metadata set

After using 64*64 as window size for segmentation, the number of samples containing cracks in the classification was about 1/15 of the total number of samples according to the result of classification,

Therefore, the image was segmented by two-scale sliding windows, and the number of images to be calculated was about 528. Compared to the original method, the proposed method greatly reduces the calculation of images.

2.2 Unsupervised clustering algorithm

Compared to the supervised learning algorithms in image classification, unsupervised learning does not require pairs of images and labels to train the model. Therefore, unsupervised learning algorithm can save a lot of resources wasted in labeling data. The data in the work was trained in the model of unsupervised clustering algorithm, thus obtaining the classification algorithm to classify the foreground and background of pavement cracks.

Clustering layer is the core of the model, but the K-Means algorithm does not perform well on high-dimensional data. Therefore, the image data was input to the cluster layer after nonlinear mapping in the work.

First, for the input of 16*16 px image data, encoder is used to extract its features, the encoder can output a feature map with shape of 2*2*32. Second, the feature map is flattened into 1 dimension, a 128-column vector can been get. Though the features for input image has been obtained, a 128-column vector is not easy for clustering layer to deal with. So, a full connection layer is added to compress it. After that, the output 10-column vector was passed to the clustering layer.

3. Rapid recognition model of pavement cracks based on unsupervised clustering

The self-encoder is a typical unsupervised learning model that takes an image as input, and outputs the reconstructed image. The self-encoder consists of encoder and decoder. Wherein, the encoder is used to extract sparse features in the input image, and the decoder uses the features output by the encoder to reconstruct the image.

3.1 Construction of encoder model

The encoder model used in the work is shown in Fig. 3. In order to ensure that the output of the encoder could be applied in the clustering layer, the convolution results were flattened. Then, the feature values were compressed using the fully connected layer to ensure that the clustering layer could obtain good results.

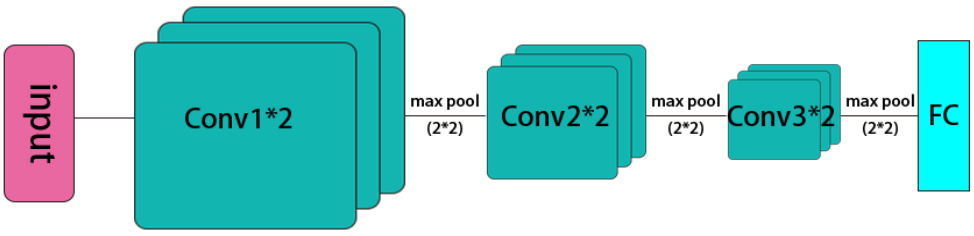

Fig. 3. Construction of encoder model

The model of the decoder is the inverse of the encoder. First, the features output by the encoder were reconstructed into two dimensions. The images were sampled using deconvolution to ensure the same size of input and output.

3.2 Implementation of clustering layer

According to Reference [12], with the soft distribution, the t-distribution is used to measure the similarity between the embedding point z_i and the center point μ_j.

$$q_{ij} = \frac{(1+\left\|z_i - \mu_j\right\|^2 / \alpha) - \frac{\alpha+1}{2}}{\sum_{j'}(1+\left\|z_i - \mu_{j'}\right\|^2 / \alpha) - \frac{\alpha+1}{2}} \tag{1}$$

Wherein, $z_i = f_\theta(x_i) \in Z$ is the embedding point corresponding to $x_i \in X$; α the degree of freedom of t-distribution; q_{ij} the probability of assigning sample i to the clustering j. As described in Reference [12], cross-validation α cannot be performed in unsupervised learning, set to 1.

For the objective function, we choose KL divergence between soft assignment q_i and auxiliary distribution p_i.

$$L = KL(P \| Q) = \sum_i \sum_j p_{ij} \log \frac{p_{ij}}{q_{ij}} \tag{2}$$

$$p_{ij} = \frac{q_{ij}^2 / f_j}{\sum_{j'} q_{ij'}^2 / f_{j'}} \tag{3}$$

Wherein, $f_j = \sum_i q_{ij}$ is the soft clustering frequency.

3.3 Deepening network

Studies show that deeper neural networks can obtain better results and more extracted features [11]. Therefore, in order to extract the features in the image, a total of six convolutional layers were used on the three-scale feature graph (See Fig. 4). A fully connected layer was added at the end of the encoder.

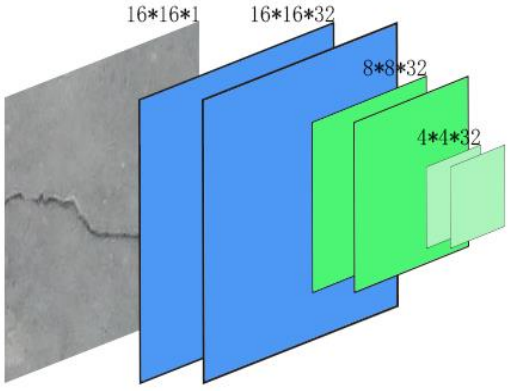

Fig. 4. Diagram of convolutional layer

By adjusting most of the hyperparameters throughout the network, the model was ensured to have good effects with fewer parameters and faster speed.

For the encoder, the work analyzed the influences of different parameter combinations on the accuracy and speed of the model. It was the most important to adjust the feature graphs extracted from each convolution layer. Two plans were mainly used (See Table 1).

At the same time, the input of the model was adjusted. The results showed that the higher resolution of the input image causes the slower speed of the model; however, the accuracy did not improve. At the same time, the resolution of all input images was adjusted to 16px only by training the model once.

3.4 Adding the Batch Normalization layer

Training Deep Neural Networks is complicated, because the fact that the distribution of each layer's inputs will change during training, because the parameters of the previous layers have changed. At the same time, as the depth of the network increases, the model will be more difficult to converge, which called gradient disappears or gradient explosion.

Batch Normalization, the batch normalization layer, is an effective way to overcome the difficulties of deep neural networks [13]. It can solve the problem of gradient in back propagation during network training.

Studies had shown that whitening the input in model training can make the network converge faster during training [14-16]. However, this method requires a large amount of computation and takes a long time. Therefore, Batch Normalization has been improved on this basis. First, batch normalization is not a joint whitening of features in the layer input and output, but independent normalization of each scalar feature to have a mean of zero and a variance of one. Second, each small batch generates an estimate of the mean and variance for each activation. In this way, the statistics used for normalization can fully participate in the back propagation of the gradient.

In the model of the work, behind the convolutional layer, a maximum BN layer was added before the maximum pooling layer to normalize the results after convolution. By adding the BN layer, the problem of gradient loss caused by the deeper encoder network is solved. It allows the encoder to achieve good feature extraction, and speeds up the convergence of the entire network.

4. Location of pavement cracks

Through the above process, the images of cracks can be determined. In this section, these images are positioned into the original images by the algorithm.

In the previous image processing, the two segmented pictures have been numbered, including the image $\{img_b \in B\}_{b=0}^{255}$ divided into 64*64, where B is the set of all the divided images. The classified images are divided into new set

$\{img_{bc} \in BC\}_{bc=0}^{n}$. A mapping from the set BC to the set B is established. The image in BC is segmented again using the sliding window to get the set $\{img_s \in S\}_{s=0}^{n*16}$. By numbering b, bc, s, the crack location can be determined.

For the set $\{img_s \in S\}_{s=0}^{n*16}$, the algorithm is used for classification to obtain the label set $\{s_i \in s\}_{i=0}^{n*16}$ of the images of cracks.

For the finally obtained number s_i, we calculate the correspondence in img_s and the image number b_i in the set B according to Equation (4):

$$b_i \leftarrow bc_i = s_i / 16 \tag{4}$$

Then the corresponding x and y coordinates are calculated according to Equation (5).

$$\begin{aligned} x &= 64 \times (b_i / 16) + 16 \times ((s_i \bmod 16) / 4) \\ y &= 64 \times (b_i \bmod 16) + 16 \times ((s_i \bmod 16) \bmod 4) \end{aligned} \tag{5}$$

In summary, the basic processing in the work is as follows.

(1) Construct a self-encoder and perform pre-training. Construct a self-encoder and pre-train the model on the data set to ensure that the encoder can correctly extract the features of the image.

(2) Construct a clustering model and train. Extract the encoder in Step 1), and connect the cluster layer to construct a clustering model. Use the dataset training model to fit the feature distribution, and form a classification model;

(3) Detect the first window sliding of the image. Use the window of scale=64 to slide the image to generate the image set B=[img1...Imgn], and zoom to 16px;

(4) Classify the models trained in the graphics application 2) in set B. Obtain the image set BC of cracks.

(5) The second window slides. Perform the window convolution of scale=16 on BC to generate the set S.

(6) Resend the images in set S to the model of 2) to obtain the final result set R;

(7) Location of the images of cracks. Locate the images of the result set R in the original images according to the algorithm in Section 4.

5. Experimental results and analysis

The data sets used in the work were from [13]. Wherein, 64px images containing pavement cracks were provided, and the positive and negative samples were 20,000, respectively. We perform 5 times for each algorithms, and get the average execution time. The data set contains road crack pictures under various conditions (including various incident angles, light intensity and other factors).

The PC's CPU using in experiment is NVIDIA gtx 950m GPU, the memory is 8 GB, and the algorithm is implemented on tensorflow.

Several controlled experiments were performed to verify the model. In the first set of experiments, we compared the accuracy of the traditional K-Means algorithm, the supervised learning algorithm and the algorithm in the work. The convolutional layer of supervised learning algorithm had the same parameters and intermediate layer as the encoder (supervised learning algorithm and the algorithm in the work have trained the same step number). The results (See Table 2) showed that although the traditional K-Means algorithm runs fast, its accuracy is only 71.12%, which cannot be used in actual processing. However, after using the deep neural network, the correctness of the classification is greatly improved, reaching the correct rate of 93.32%. The algorithm in the work is 0.81% higher than that in Reference [17]. Although there is still a gap in accuracy compared with the supervised learning algorithm, it has faster speed, better practicability, and more applications, more importantly, no markup is required.

Table 1

Hyperparameters used by the two models in the convolutional layer

Model	Conv1	Conv2	Conv3
A	32	64	128
B	32	32	32

Table 2

Comparison of accuracy between traditional clustering algorithm and the algorithm in this paper

Algorithm	Samples	Correct identification number	Accuracy	time
The traditional K - Means	40000	28450	71.12%	0.162s
Supervised learning algorithm	40000	39839	99.59%	4.001s
References [17]	40000	37004	92.51%	unmeasured
The proposed algorithm	40000	37328	93.32%	4.356s

In the second set of comparative experiments, the work compared the effects of 64*64 image blocks and 16*16 image blocks on the accuracy and speed of the proposed algorithm. In the experiment, 5,000 samples were randomly selected from the test samples to calculate the running time and accuracy of the algorithm. The experimental results (See Table 3) showed that when using 64*64 and 16*16 images as the input of the model, the impact on the accuracy of the results was not large, with the difference of 0.44%. However, with the 64*64 image as the input to the model, the speed of the model became very slow. It cost 4.156 s to calculate 5,000 sheets of data. Compared with the cost time of 1.611 s with 16*16 images as input, the speed of the latter has increased more than 2 times, the recognition results are shown in Fig. 5.

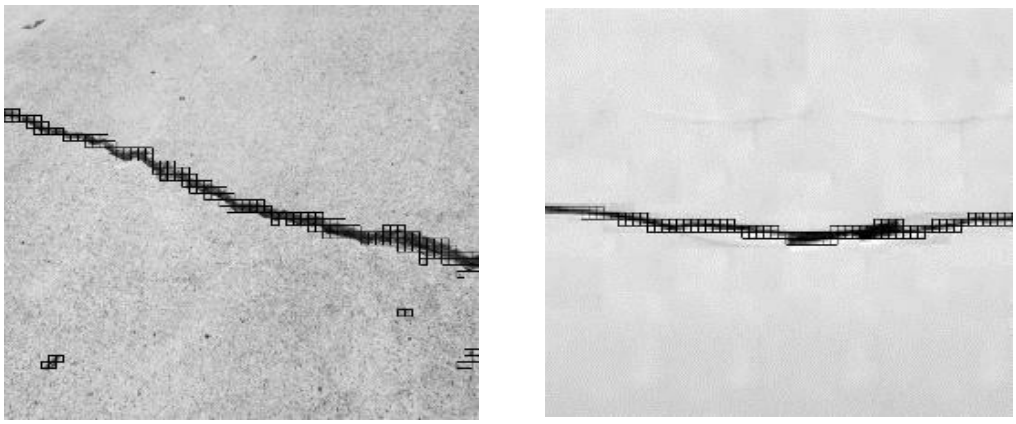

Fig. 5. Detection and positioning results

Table 3

Performances of the algorithm in 64px and 16px images

Resolution	Correct identification number	Accuracy rate	Run time
64px	4704	94.08%	4.156s
16px	4682	93.64%	1.611s

In the third set of experiments, the work compared the effects of the hyperparameters on the model. In model A, the super-parameters were adjusted to extract more feature graphs. In model B, the same hyperparameters were used in different convolution layers. Please see Table 1 for more information about the super-parameters. The results (See Tables 1 and 4) showed that the parameters that model A needed to learn far exceeded those that model B needed, but model B had shorter running time and better accuracy. The reason is that in order to ensure that the clustering layer can fit the distribution of features, the last few layers in the encoder greatly compress the dimensions of features. Model A extracts more features in the convolutional layer, which are lost in the last few layers.

Table 4

Effect of the two models on the results

Model	Number of parameters	Accuracy rate	Run time
A	339658	89.53%	5.445s
B	48254	93.32%	4.251s

In the fourth set of experiments, the accuracy and speed of the model were compared with DBCC. In the experiment, we randomly selected 5,000 samples from the test samples. The results (Table 5) show that DBCC has a higher accuracy rate of 97.34%, while our model has an accuracy of only 93.32%. Both

models are very fast, but our models are faster. But the results are based only on the deep learning model. Then tested the speed of the entire algorithm. The results (Table 6) show that our algorithm has a better speed than DBCC because the multi-scale sliding window algorithm reduces performance waste.

Table 5

Comparison of accuracy between our model and DBCC

Model	Correct identification number	Accuracy rate	Run time
The proposed algorithm	4666	93.32%	1.532s
DBCC	4867	97.34%	1.732s

Table 6

Comparison of speed between our algorithm and DBCC

Algorithm	Number of images	Run time
The proposed algorithm	10	0.982s
DBCC	10	1.638s

6. Conclusions and discussion

The work proposed a new fast detection algorithm of pavement cracks based on unsupervised learning and discussed the application of unsupervised learning in practice. Besides, we introduced improvement of the algorithm in various aspects and the adjustment of hyperparameters of models in detail. The simple t-distribution was used to achieve the higher accuracy of the clustering algorithm. At the same time, the algorithm was greatly accelerated by optimizing parameters. The recognition accuracy of the proposed algorithm was 0.81% higher than the existing supervised learning algorithm, although there was still a gap in accuracy compared with the latter. However, in the actual processing, it has faster speed to mark, better practicability, and more applications.

The future research focuses on improving the accuracy of the algorithm and the generalization of the model by adjusting the probability distribution used by the clustering layer. At the same time, we will explore the end-to-end realization of the detection system of pavement cracks based on deep learning algorithm.

Acknowledgements

The author was sponsored by the National Natural Science Funds, Key Laboratory of Geoscience Spatial Information Technology of Ministry of Land and Resources, Chengdu University of Technology, China funds Sichuan science and technology innovation miaozi project. The author acknowledge the National Natural Science Funds (NO. 41372340), Key Laboratory of Geoscience Spatial

Information Technology of Ministry of Land and Resources, Chengdu University of Technology, China funds (NO. KLGSIT2016-10), Sichuan science and technology project (NO. 2019JDPT0014).

REFERENCES

[1]. *X. Huang*, "Automatic inspection of pavement cracking distress", in Journal of Electronic Imaging, vol. 15, 2006, pp. 13-17.

[2]. *Y. Fei, K. Wang, A. Zhang, et al.,* "Pixel-Level Cracking Detection on 3D Asphalt Pavement Images Through Deep-Learning-Based CrackNet-V", in IEEE Transactions on Intelligent Transportation Systems, vol. 99, 2019, pp. 1-12.

[3]. *F. Roli,* "Measure of texture anisotropy for crack detection on textured surfaces", in Electronics Letters, vol. 32, 1996, pp. 1274-1275.

[4]. *L. Jing, Z. Aiqin,* Pavement crack distress detection based on image analysis, 2010 International Conference on Machine Vision and Human-machine Interface. IEEE, 2010, pp. 576-579.

[5]. *P. Cisar, S. M. Cisar, B. Markoski,* Kernel sets in compass edge detection, Computational Intelligence and Informatics (CINTI), 2013 IEEE 14th International Symposium on. IEEE, 2013, pp. 239-242.

[6]. *Y. Shi, L. Cui, Z. Qi, et al.,* "Automatic road crack detection using random structured forests", in IEEE Transactions on Intelligent Transportation Systems, vol. 17, 2016, pp. 3434-3445.

[7]. *S. Z. A. Rahman, S. N. H. S. Abdullah, K. A. Z. Ariffin*, "Gait Recognition based on Inverse Fast Fourier Transform Gaussian and Enhancement Histogram Oriented of Gradient", in International Journal on Advanced Science, Engineering and Information Technology, vol. 8, 2018, pp. 1402-1410.

[8]. *T. Takahashi, M. Ogawa, A. Sato, et al.,* "Translocated duplication of a targeted chromosomal segment enhances gene expression at the duplicated site and results in phenotypic changes in Aspergillus oryzae", in Fungal biology and biotechnology, vol. 5, no. 1, 2018, pp. 17.

[9]. *K. He, X. Zhang, S. Ren, et al.,* Deep residual learning for image recognition, Proceedings of the IEEE conference on computer vision and pattern recognition, 2016, pp. 770-778.

[10]. *C. Szegedy, W. Liu, Y. Jia, et al.*, "Going deeper with convolutions", in Computer vision and pattern recognition, vol. 1, 2015, pp. 1-9.

[11]. *L. F. Li, W. F. Ma, et al.,* "Research on Detection Algorithm for Bridge Cracks based on Deep Learning", in ACTA AUTOMATICA SINICA, vol. 44, 2018, pp. 1-14.

[12]. *J. Xie, R. Girshick, A. Farhadi,* Unsupervised deep embedding for clustering analysis, International conference on machine learning, 2016, pp. 478-487.

[13]. *L. Zhang, F. Yang, Y. D. Zhang, et al.,* Road crack detection using deep convolutional neural network, Image Processing (ICIP), 2016 IEEE International Conference on. IEEE, 2016, pp. 3708-3712.

[14]. *Y. Lecun, L. Bottou, G. B. Orr, et al.,* Efficient BackProp, Lecture Notes in Computer Science, 1998, pp. 9-50.

[15]. *D. V. Howard, S. E. Goldin,* "Selective processing in encoding and memory: An analysis of resource allocation by kindergarten children", in Journal of Experimental Child Psychology, vol. 27, 1979, pp. 0-95.

[16]. *J. H. Olejarczyk, S. G. Luke, J. M. Henderson,* "Incidental memory for parts of scenes from eye movements", in Visual Cognition, vol. 22, 2014, pp. 975-995.
[17]. *W. Liu, D. Anguelov, D. Erhan, et al.,* SSD: Single shot multibox detector, European conference on computer vision. Springer, Cham, 2016, pp. 21-37.