# Fish Detection and Classification Using Convolutional Neural Networks

**Chapter** · January 2020

DOI: 10.1007/978-3-030-37218-7_128

Rekha B S
Rashtreeya Vidyalaya College of Engineering
**6** PUBLICATIONS **14** CITATIONS
SEE PROFILE

Dr. Srinivasan G N
Rashtreeya Vidyalaya College of Engineering
**32** PUBLICATIONS **400** CITATIONS
SEE PROFILE

Sravan Reddy
Rashtreeya Vidyalaya College of Engineering
**1** PUBLICATION **12** CITATIONS
SEE PROFILE

Divyanshu Kakwani
Indian Institute of Technology Madras
**5** PUBLICATIONS **137** CITATIONS
SEE PROFILE

# Fish Detection and Classification Using Convolutional Neural Networks

B. S. Rekha[1](✉), G. N. Srinivasan[2], Sravan Kumar Reddy[3],
Divyanshu Kakwani[3], and Niraj Bhattad[3]

[1] Bharathiar University, Coimbatore, Tamil Nadu, India
rekhabs@rvce.edu.in

[2] Department of Information Science and Engineering,
R.V. College of Engineering, Bengaluru, India
srinivasangn@rvce.edu.in

[3] R.V. College of Engineering, Bengaluru, India
sravankr96@rvce.edu.in, divkakwani@gmail.com

**Abstract.** About fifty percent of the world relies on seafood as main protein source. Due to this, the illegal and uncultured fishery activities are proving to be a threat to marine life. The paper discusses a novel technique that automatically detects and classifies various species of fishes such as dolphins, sharks etc. to help and protect endangered species. Images captured through boat-cameras have various hindrances such as fluctuating degrees of luminous intensity and opacity. The system implemented, aims at helping investigators and nature conservationists, to analyze images of fishes captured by boat-cameras, detect and classify them into species of fishes based on their features. The system adapts to the variations of illumination, brightness etc. for the detection process. The system incorporates a three phase methodology. The first phase is augmentation. This phase involves using data augmentation techniques on real time images dataset captured by boat-cameras and is passed to the detection module. The second phase is the detection. This phase involves detecting fishes in the image by searching for regions in the image having high probability of fish containment. The third phase is the classification of the detected fish into its species. This step involves the segmented image of fishes to be passed to the classifier model which specifies to which species the detected fish belongs to. CNN (Convolutional neural network) is used at the detection and classification phase, with different architectures, to extract and analyze features. The system provides confidence quotients on each image, expressed on a 0–1 scale, indicating the likelihood of the image belonging to each of the following eight categories ALB, BET, YFT, LAG, DOL, Shark, Other and None. The system provides detection and classification with an accuracy of 90% and 92% respectively.

**Keywords:** Object detection · Classification · Computer vision · Deep learning · Convolutional neural network

S. Smys et al. (Eds.): ICCVBIC 2019, AISC 1108, pp. 1221–1231, 2020.
https://doi.org/10.1007/978-3-030-37218-7_128

# 1 Introduction

In order to preserve marine ecosystem, real-time inspection of fisheries is necessary. In coastal areas, where most of the fishes are caught, uncultured practices are proving to be detrimental to marine ecosystem. For the conservation of fish species, the study of size and diversity of each of such species is vital. There has been a lack of automatic methods to perform real-time detection and classification of captured fishes. Traditionally, this task has been achieved by doing manual inspection. This requires having an expert on-board who determines fish species at the time of capture. Other methods involve broadcasting the boat snapshots to remote experts who determine the fish species that the captured fish belongs to.

Each of these methods has their drawbacks. First, having a fish inspector at every boat is not feasible. Second, it is difficult to find experts in the field of ichthyology. Third, transmitting boat snapshots to remote experts is vulnerable to failures. With the nature of human beings turning out to be destructive towards marine eco-system, the conservation of fishes becomes an important aspect. Thus a system which analyses the images of fishes captured by boat cameras and automatically detects and classifies them into their type of species acts as a key to solve such a problem. The current system receives real time input images from a camera installed aboard boats. Its task is to detect and classify the fishes in these images into one of the following 8 categories - ALB, BET, YFT, LAG, DOL, Shark, Others and no fish. The images in the available dataset are shot with varying imaging conditions. These images are highly distorted, cluttered, and in several of them, the fishes are partly occluded. This makes the task of detection and classification challenging. The current state-of-the-art algorithms using the traditional image processing techniques do not perform very well on such images and also has proved to be very specific with the type of environment.

# 2 Related Work

The system has to be independent of the environment and reliable in the case of noise. This can be achieved by using Deep Learning techniques which involve Convolutional Neural Network (CNN) in the detection and classification tasks. This technique was introduced first by Alex as ALEXNET in the ImageNet competition which gave remarkable results compared to the previous approaches [4]. Similar to all the machine learning algorithms, the method had a training phase and a testing phase. In training phase, the pipeline consists of three parts, data augmentation, convolution network and a fully connect layer at the end combined with a softmax activation. Each part of this design has a specific function. The main function of convolutional layers is to extract features from the image. Each convolutional layer has certain number of filters. These filters are weighted kernels that are passed through out the image to generate corresponding feature maps. These feature maps are passed on to the fully connected network. The fully connected layer weights the features in the final feature maps generated and gets a final probability distribution over all the possible classes. Several hyper parameters are used to tune the network model which include error correction method, learning rate, number of filters in each convolutional layer, number of convolutional layers, input image size etc. Using deeper networks results in better accuracy can be achieved when combined with the techniques like dropout and skip networks which are introduced in [5, 6]. A dropout is the technique where a fraction of connections in the fully connected layers are desperately removed in a random order. A clear explanation is given by the Alex team in [6]. Overfitting is the condition where the grading moves from least error point towards high error rate. This generally happens due to over training a network with same dataset. A simple indication for this is the training accuracy keeps increasing whereas the validation accuracy shows a gradual decrease. The Skips nets also uses a similar approach where few convolutional layers will be desperately skipped and the feature maps are passed on to the further layers. The main idea behind this is to compute a feature map based on two feature maps, one from the direct parent and another from the ancestor. This experimentally showed that the approach reduces overfitting of the network. Using an Ensemble of different models is another technique which gives an edge over using a single model prediction is another popular technique which is used in most of the modern deep learning problems [4, 6–8]. Different approaches like using parallel convolutional neural networks are used by the Google research team [7] which showed improved results but uses a very complex architecture.

Since the success of AlexNet [4], several detection approaches based on CNNs have been proposed. One such successful technique combined region proposals with CNNs, giving rise to R-CNN. The initial version of R-CNN is described in [9]. Several variants of the R-CNN in [9] have been proposed since then. The most successful among them are: Fast R-CNN [10], Faster R-CNN [11], DeepBox [12], R-FCN [13].

The R-CNN described in [9] first generates candidate regions using Selective Search [14], and for each region, generates a feature vector using a CNN, and finally uses an SVM per each category to assign category scores to each region proposal.

R-CNNs were later improved by changing the region proposal generation algorithm, for instance Edgeboxes [15] is used in [16]. For a comparison of region proposal algorithm, see [17]. DeepBox [12] refined the region proposal algorithm by adding a shallow CNN network to filter the candidates. A better version of R-CNN, called Fast R-CNN, was introduced in [10]. It reduced the time by training in single-stage and by sharing the computation of feature vector corresponding to each region proposal. The techniques based on region proposal generation, however, fail in case of cluttered images. In such cases, it is difficult to extract meaningful regions out of an image, mainly because of high-object density and occlusion rates.

## 3 System Architecture

The proposed system consists of three phase – Data augmentation, detection and classification. As the initial dataset is small and highly imbalanced, the images were augmented to produce a sizeable balanced dataset. Various augmentation techniques were used to increase the size of the data set. In the detection stage, the input image is segmented to obtain image patches that contain fish. The obtained patches are passed to the classifier. The classifier produces a probability distribution of the fish classes of the detected fish (Fig. 1).

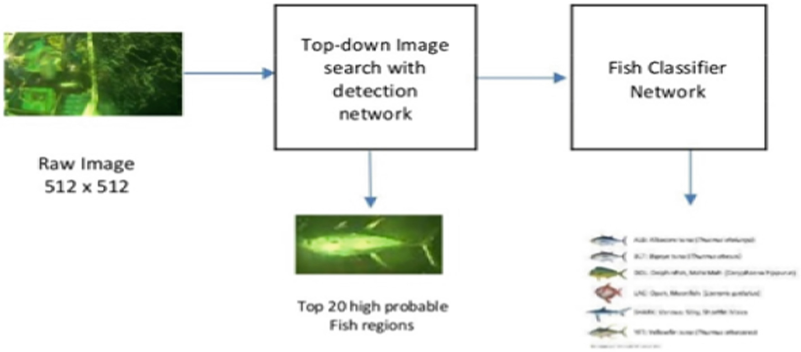
Raw Image
512 x 512
Top-down Image search with detection network
Top 20 high probable Fish regions
Fish Classifier Network
ALB: Albacore tuna (Thunnus alalunga)
BET: Bigeye tuna (Thunnus obesus)
DOL: Dolphinfish, Mahi Mahi (Coryphaena hippurus)
LAG: Opah, Moonfish (Lampris guttatus)
SHARK: Various: Silky, Shortfin Mako
YFT: Yellowfin tuna (Thunnus albacares)

**Fig. 1.** System architecture.

### 3.1 Augmentation

Augmentation phase is used to enhance the training dataset in terms of both quality and quantity. Simple affine transforms and preprocessing techniques are used to achieve that.

### 3.2 Detection

The fishes present in the input image are detected in this phase. Different techniques are used to find an optimal approach to extract the fishes from the image. Initially, localization technique was applied in which, the input image is fed to a localization network which regresses coordinates for the area within which fish is present. The limitation of regression bounding is that it cannot detect multiple fishes in a single image. As the dataset consists of multiple fishes in a single image, a better method to find more than one fish was required, hence a custom detection algorithm that best matches the dataset is designed and implemented (Fig. 2).

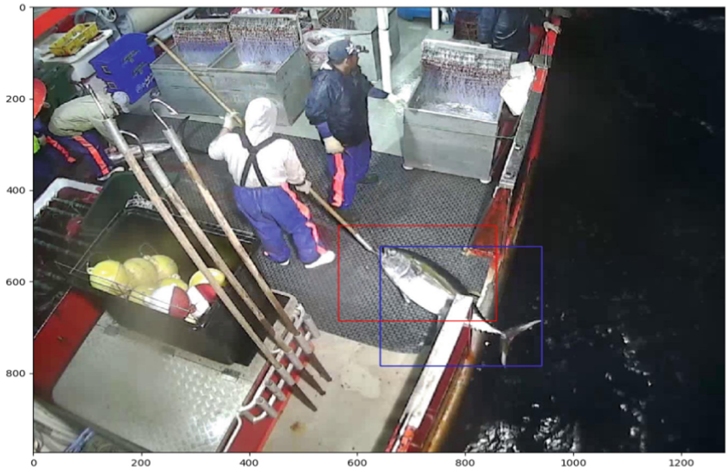

**Fig. 2.** Output of the localization algorithm [6]

The method used is top down image search, where the fishes are detected in two levels. The initial level detection is a loose bound of the fish and the latter is a tight bound. At each level different networks are used, whose input is a patch which will be classified into fish or no-fish. The patches are extracted from the raw image in the dataset with different window sizes that are selected in two ways. One is applying k-means clustering on the set of window shapes containing the fishes in the annotated dataset. The other is to select a range of windows with a constant interval. In the proposed system architecture, the latter method is chosen to generalize the system. These obtained window sizes are used to extract the patches for detection phase (Fig. 3).

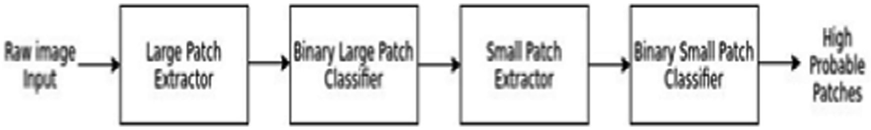

**Fig. 3.** Detection algorithm flow diagram.

The detection pipeline consists of a large patch extractor, binary classifier, small patch extractor followed by another binary classifier. The initial large patch extractor extracts large square patches ranging from a scale of 300 × 300 to 600 × 600 with an interval of 50. These patches are classified with a binary classifier that is trained with similar image patches manually extracted from the dataset. The binary classifier also outputs the probability of each image patch containing a fish. Based on the classifier output, top 10 high probable image patches are selected and forwarded to next level segmentation. Smaller image patch extractor extracts small patches ranging from a scale of 100 × 100 to 300 × 300 with an interval of 50. These patches are classified with a binary classifier that is trained with similar fish patches manually extracted from the dataset. Moreover, computation time depends on model and image crop size, but precision is also affected; usually, time and precision have trade-off relation (Fig. 4).

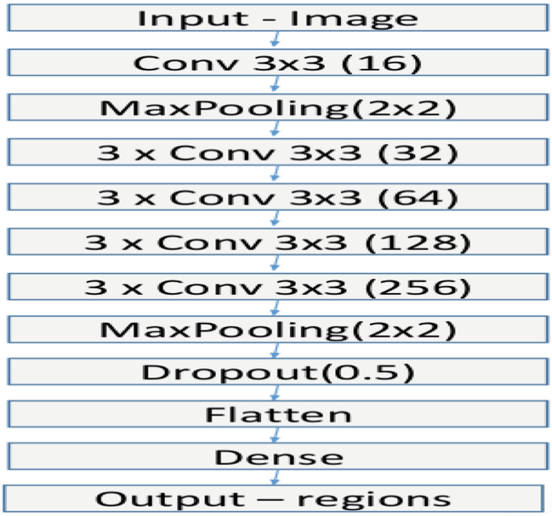

**Fig. 4.** Architecture of the detection network

The detection phase uses the top down image search technique with sliding window approach in order to initially detect the presence of the fish or no-fish. Figure 5 shows the presence of a fish.

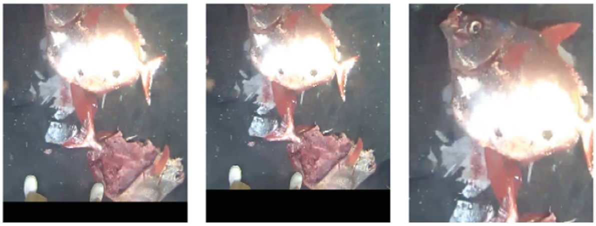

**Fig. 5.** Output of the detection algorithm.

### 3.3 Classification

All the input patches obtained in the detection phase are resized appropriately passed to the classifier network. For each input patch, the classifier returns a probability distribution of fish classes. The final probability distribution is computed as follows:

$$\begin{aligned} &P(x) = max\ (pi\ (x)),\ if\ x\ != \text{'NoF'} \\ &avg\ (pi\ (x)),\ otherwise \end{aligned} \tag{1}$$

The Classification network consisted of several convolutional layers followed by a fully connected (FC) layer. VGG-16 network is used for classifier with the input size of (224, 224, 3). This network was chosen because pre-trained VGG networks are available and since our dataset is small, a pre-trained model is indispensable (Fig. 6).

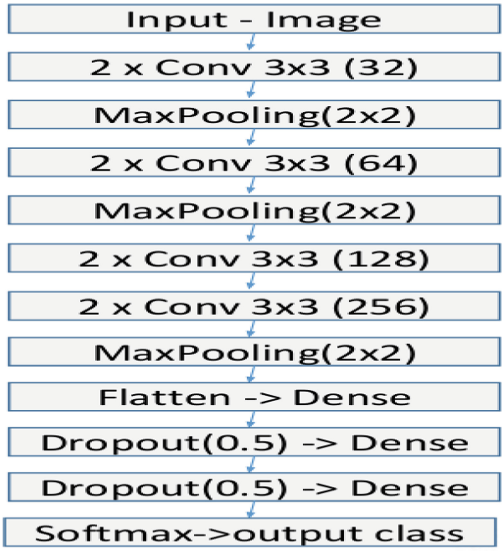

**Fig. 6.** Architecture of the classification network.

Once the images pass thru the detection phase are passed thru the classification phase. In the classification phase, if the fish is detected the classifier classifies the detected fish into one of the eight categories. The class with the highest probability is conjectured to be the class of the fish contained in the image. Figure 7 shows the classification of a fish detected in the input image to be of the category BET with a high probability of 1.0. Figure 8 shows that there is No-Fish on the board with a high probability of 1.0.

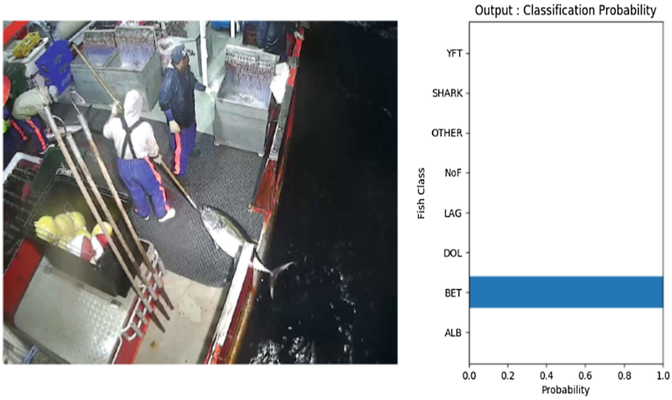

**Fig. 7.** Output of the classification algorithm identifying BET fish

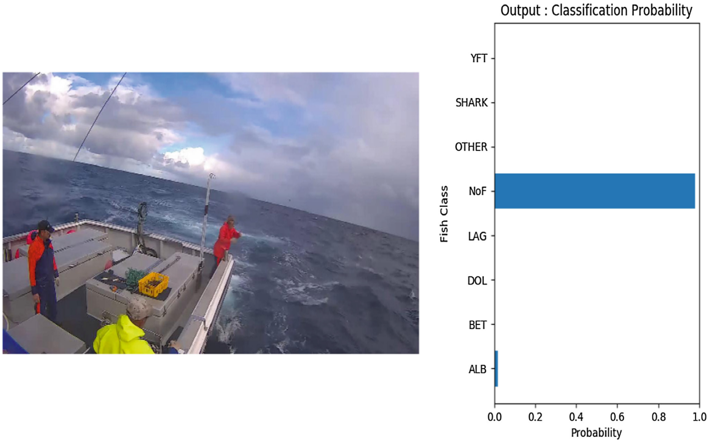

**Fig. 8.** Output of the classification algorithm identifying No fish [7]

## 4 Dataset

The original dataset is provided by National Conservancy Organization, who in turn extracted it from camera feeds obtained from boat cameras. The dataset contains a total of 3777 images divided into 8 categories, namely ALB, BET, YFT, SHARK, LAG, DOL, NoF, Other. Since the dataset is relatively small, and contains plenty of variances in imaging conditions, the task of classification is challenging. Following are some of the challenges faced during processing such images:

- Different boat environments: Different environments contain different camera position and orientation.
- Different capture times: Some images are taken in day-light, while some are taken in night-time.
- Distortion: Many images are distorted due to the general hustle typically present in fisheries.
- Occlusion: Several images contain fishes occluded by other objects or fishermen.

Prior to training the networks, the available dataset is heavily augmented with rotation and Gaussian blur. The initial dataset contained 3777 images, and augmentation process resulted in 16000 images. The augmented dataset is fully balanced across all the categories. This dataset is split into training and validation in the ratio of 8:2. Data augmentation increases the data set size which provides better training and also helps in overcoming the neural net getting over fit.

## 5 Training

In training phase, the pipeline consists of three parts, data augmentation, convolution network and a fully connect layer at the end combined with a softmax activation. Each part of this design has a specific function. The main function of convolutional layers is to extract features from the image. Each convolutional layer has certain number of filters. These filters are weighted kernels that are passed through out the image to generate corresponding feature maps. These feature maps are passed on to the fully connected network. The fully connected layer weights the features in the final feature maps generated and gets a final probability distribution over all the possible classes. Several hyper parameters are used to tune the network model which include error correction method, learning rate, number of filters in each convolutional layer, number of convolutional layers, input image size etc.

The detection and classification networks are trained with the following parameters:

- Optimizer: Adams
- Learning Rate: 1e−3
- Objective: Categorical Cross-entropy
- Epochs: 40

The classifier uses a VGG-16 network pre-trained on the ImageNet dataset, and fine-tuned on tight patches of fishes extracted from our training data. 2000 patches are extracted for each category from the training dataset. For categories that do not have enough samples, data augmentation techniques are used.

## 6 Results

Initially a single network was used that performed the classification job. The Training Accuracy was 95%, and the Validation Accuracy was poor. When this initial model was tested it was found that there was a dependency on the environment which made us to introduce a detection phase before classification.

The initial approach for detection was localization of the fish in the image. This approach failed in the case where there are multiple fishes in a single image as the localization task was meant to find only one object of interest in an image. This made us to evolve a new method of detecting all the fishes in an image using an image search algorithm. With this final design of network architectures and training parameters, Following are the results obtained:

Detection
Training Accuracy: 94%
Validation Accuracy: 90%
Training Time: About 6 h
Test Time: 5 s
Classification
Training Accuracy: 96%
Validation Accuracy: 92%
Training Time: About 3 h
Test time: 50 ms

## 7 Conclusion

The usage of CNNs in the detection and classification produced remarkable results. It gave significantly better results than the methods based on manual feature extraction and the traditional image processing techniques. The final networks used are derived from the VGGNet. The CNNs used in the detection module and the classification module achieved a validation accuracy of about 90% and 92% respectively. While the accuracy of the networks has been remarkable, the whole process is a little slow. The detection module takes about 5 s to process the image, while the classification module is almost instantaneous, taking less than a second. The training of the networks has been done on limited datasets. The usage of better aggregation tools could produce much better results in production. The future scope incudes working on R-CNN for better results in the detection and classification phases. The project could be extended to detecting and classification of other species of fishes also.

**Compliance with Ethical Standards**

✓ All authors declare that there is no conflict of interest.

✓ No humans/animals involved in this research work.

✓ We have used our own data.

## References

1. Lowe, D.G.: Distinctive image features from scale-invariant key-points. IJCV **60**, 91–110 (2004)
2. Clara Shanthi, G., Saravanan, E.: Background subtraction techniques: systematic evaluation and comparative analysis. Int. J. Mod. Eng. Res. (IJMER) **3**, 514–517 (2013)
3. Jones, V.: Rapid object detection using a boosted cascade of simple features. In: Computer Vision and Pattern Recognition (2001)
4. Krizhevsky, A., et al.: ImageNet classification with deep convolutional neural networks. In: NIPS (2012)
5. He, K., et al.: Deep residual learning for image recognition. ISLR (2015)
6. Alex, et al.: Dropout: a simple way to avoid overfitting in the network. JMLR (2014)
7. Szegedy, C., Liu, W., et al.: Going deeper with convolutions. In: CVPR. Google Research (2015)
8. Simonyan, K., Zisserman, A.: Very deep convolutional networks for large scale image recognition. In: ICLR. Visual Geometry Group, Department of Engineering Science, University of Oxford (2015)
9. Girshick, R., Donahue, J., Darrell, T., Malik, J.: Rich feature hierarchies for accurate object detection and semantic segmentation. In: The IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 580–587 (2014)
10. Girshick, R.: Fast R-CNN. In: The IEEE International Conference on Computer Vision (ICCV), pp. 1440–1448 (2015)
11. Ren, S., He, K., Girshick, R., Sun, J.: Faster R-CNN: towards real-time object detection with region proposal networks. In: Advances in Neural Information Processing Systems 28, NIPS (2015)
12. Kuo, W., Hariharan, B., Malik, J.: DeepBox: learning objectness with convolutional networks. In: The IEEE International Conference on Computer Vision (ICCV), pp. 2479–2487 (2015)
13. Dai, J., Li, Y., He, K., Sun, J.: R-FCN: object detection via region-based fully convolutional networks. In: Advances in Neural Information Processing Systems 29, NIPS 2016 (2016)
14. Uijlings, J.R.R., van de Sande, K.E.A., Gevers, T., Smeulders, A.W.M.: Selective search for object recognition. Int. J. Comput. Vis. **104**(2), 154–171 (2013)
15. Lawrence Zitnick, C., Dollr, P.: Edge boxes: locating object proposals from edges. In: European Conference on Computer Vision, ECCV 2014, pp. 391–405 (2014)
16. Tang, S., Yuan, Y.: Object detection based on convolutional neural network, Stanford Project report (2016)
17. Hosang, J., Benenson, R.: How good are detection proposals, really? Computer Vision and Pattern Recognition, arXiv (2014)
18. Gidaris, S., Komodakis, N.: LocNet: improving localization accuracy for object detection, arXiv (2016)
19. Sermanet, P., Eigen, D., Zhang, X., Mathieu, M., Fergus, R., LeCun, Y.: OverFeat: integrated recognition, localization and detection using convolutional networks, arXiv (2014)

20. Jaderberg, M., Simonyan, K., Zisserman, A., Kavukcuoglu, K.: Spatial transformer networks, arXiv (2016)
21. Redmon, J., Divvala, S., Girshick, R., Farhadi, A.: You only look once: unified, real-time object detection. In: CVPR (2015)
22. Kingma, D.P., Ba, J.L.: Adam: a method for stochastic optimization. In: ICLR (2015)
23. Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., Salakhutdinov, R.: Dropout: a simple way to prevent neural networks from overfitting. J. Mach. Learn. Res. **15**, 1929–1958 (2014)
24. Ioffe, S., Szegedy, C.: Batch normalization: accelerating deep network training by reducing internal covariate shift, arXiv (2015)
25. Liu, W., Anguelov, D., Erhan, D., Szegedy, C.: SSD: Single Shot MultiBox Detector, arXiv (2016)
26. Mnih, V., Heess, N., Graves, A., Kavukcuoglu, K.: Recurrent models of visual attention. In: ICLR. Google DeepMind (2016)
27. Liang, M., Hu, X.: Recurrent convolutional neural network for object recognition. In: CVPR (2015)