An improved YOLOv3 model based on skipping connections and spatial pyramid pooling

Xinliang Zhang, Wanru Wang, Yunji Zhao and Heng Xie

School of Electrical Engineering and Automation, Henan Polytechnic University, Jiaozuo, People’s Republic of China

ABSTRACT
The cascaded deep-learning network of YOLOv3 emphasizes on the layer-wise feature extraction. It neglects the sequential influence among the layers that contributes to the subtle features for the objects detection. An improved YOLOv3 model with skipping connections is proposed in this paper for the sufficient utilization of layer-wise features. Firstly, a DenseBlock network is adopted as the fourth and fifth down-sampling layers of YOLOv3. The DenseBlock is characterized of a parallel architecture and capable of transmitting the features backwards among extraction layers. Then, the features of preceding layers are incorporated by a skipping fashion into subsequential layers. Secondly, a spatial pyramid pooling (SPP) module is introduced at the neck of the object detector. It realizes the size-tuning of the model input. Then the multi-scaled region features are generated after the pooling and concatenation operation of the SPP. Finally, the validation experiments have been conducted on a dataset of helmet objects. The results have shown that the proposed YOLOv3 model improves the accuracy effectively. It yields a mean average precision 88.6% on the helmet detection, which is 3.5% higher than the original YOLOv3 network.

1. Introduction
Most classical object detection methods relied on the colour histogram information, texture features and etc. (Fei-Fei & Perona, 2005). The AlexNet model showed a high detection accuracy on the ImageNet dataset, which attracted many scholars to study object detection based on the deep learning (Krizhevsky et al., 2012). In general, the deep learning model has two kinds of structure. Two-stage networks take sliding window and efficient selective search to get region proposal such as FastR-CNN (Ross, 2015) and Faster R-CNN (Ren et al., 2017). While SSD (Liu et al., 2016) and a series of YOLO (Redmon & Farhadi, 2017, 2018; Redmon et al., 2016) models are typical representatives of one-stage networks without the usage of the region proposal. Specially, YOLO advances to get the detection result in one-step. It directly generates target information without region proposal, including the classification, precision and position coordinates. The SSD model continues the YOLO’s idea and integrates the multi-scaled feature adaption and detection of small targets (Liu et al., 2016). Further, YOLOv3 model (Redmon & Farhadi, 2018) uses the Darknet-53 as the backbone network. It provides a rather higher mAP than SSD and YOLO models with the ResNet and multi-scale detection. From YOLOv1 to YOLOv3, it can be seen that the parallel structure with the ResNet (He et al., 2016) and FPN (Lin et al., 2017) contributes to the accuracy improvement of the detection model. Yunong T changed the YOLOv3 backbone network to a dense connection for the combination of features from different layers (Tian et al., 2019). In Zhou et al. (2018) the DenseNet-169 was adopted as the backbone network of STDN to enhance the parallel relation. With the enhancement of feature transmission between layers, the model had yielded a significant improvement on the detection accuracy. At the same time, it is noted that the spatial pyramid pooling network (SPPNet), widely used in multi-scale scenes (Gao et al., 2017; He et al., 2015; Lecun et al., 2015; Özdemir et al., 2010), equivalently adds a parallel structure to the cascade network. SPPNet imposed different sized convolution kernels at the same layer. It enlarged the receptive fields of the network by concurrent use of the features. In Liu et al. (2020), the atrous spatial pyramid pooling (ASPP) block utilized different rated dilation convolutions in parallel to capture multi-scale features. It achieved a high segmentation performance. In this paper, the skipping connections are introduced into the YOLOv3 to improve the utilization of features among layers. Meanwhile, a SPP is adopted...
to enhance the adaption of the model to multi-sized features.

The remainder of the paper is organized as follows. Section 2 details the method of YOLOv3 with a comparison to YOLOv1. Section 3 describes the realization of the skipping connections and Section 4 the operation of involved SSP block. Then, the improved YOLOv3 model is presented in Section 5. And the validation results are demonstrated in Section 6. Finally, Section 7 gives the concluding remarks and discusses possibilities for future work.

2. Methodologies

The comparison of YOLOv1 and YOLOv3 detection models is given in Figure 1. YOLOv1 as shown in Figure 1(a), the convolution layer and pooling layer are stringed together to form the backbone network. It takes a head–tail feedforward structure as a whole, i.e. the output of the current extraction layer is exactly the input of the next layer. The input of the current layer does not act as any input component and exerts no direct effect on the next layer. YOLOv1 leads to an insufficient utilization of feature extraction. The unexpected gradient disappearance would happen due to the non-smoothness of the actuation function during the model training. Compared with YOLOv1, YOLOv3 takes Darknet-53 as the backbone network. As shown in Figure 1(b), YOLOv3 improves the network by inserting a ResBlock at the frontend of Darknet-53. It is a parallel structure. It deepens the network structure and avoids the gradient disappearance. Meanwhile, YOLOv3 changes the network to multi-scale detection at the backend. There are 5 down-sampling blocks in the Darknet-53 to achieve the scale diversity. Each down-sampling block is composed of 3 \times 3 convolutional layers. Because of the scale diversity, the feature maps contain both general information and semantic information.

However, YOLOv3 implements down sampling through a convolutional layer in series. Without fusion of the current feature map into each subsequent layer, it leads to a deficiency in the extraction of multi-layer semantic information in the process. Moreover, the receptive field become rather limited because of the use of small-sized convolution kernels to pool the current layer.

3. Skipping connections

Densely connected convolutional networks can realize connecting each layer to every other layer in a feed-forward fashion. More feature maps of inconsistent dimensions can be obtained through skipping connections. The operation of the DenseBlock is shown in Figure 2. The involved feature maps contain both shallow detail features and deep semantic features naturally. The output feature map is described as

$$x_l = H_l(x_0 \oplus x_1 \oplus \ldots \oplus x_{l-1})$$

where \(x_i, i = 0, 1 \ldots l-1\), is the corresponding output of the layer \(i\); \(\oplus\) is the concatenation operator for the feature maps; \(H_l\) uses the function BN-ReLU-Conv(1 \times 1)-BN-ReLU-Conv(3 \times 3), which is composed of the batch normalization (BN), rectified linear units (ReLU) and the convolution network(Conv).

As shown in Figure 2, each layer takes all preceding feature maps as its input. The feature map \(x\) sized of \(W \times H \times C\) initially passes through a convolutional layer with a convolution kernel of \(3 \times 3\) and stride of \(2\), but the number of filters is half of the original’s. Subsequently, the derived feature map \(x_0\), sized of \(W/2 \times H/2 \times C\), inputs \(H_1\) to generate the feature map \(x_1\). Through the skipping connections, the input of \(H_2\) is a concatenation of preceding feature maps \(x_0\) and \(x_1\), i.e. \(x_0 \oplus x_1\). The latter operation is similar. Finally, the output of the DenseBlock network with \(l\) feature layers is \(x_0 \oplus x_1 \oplus \ldots \oplus x_l\).

The computation consumption of a network is defined as the Billion Floating point operations (BFLOPs), i.e.

$$C = (2.0 \times l_0 \times l_{size} \times l_{size} \times l_c/l_{groups} \times l_{out,h} \times l_{out,w}) / 10^9$$

where \(l_0\) and \(l_c\) are the numbers of convolution kernels and filters within the current layer; \(l_{size}\) is the size of convolution kernel; \(l_{out,h}\) and \(l_{out,w}\) represent the size of the feature map after convolution. The default value of \(l_{groups}\) is 1.

For original model in Figure 1(b), it leads

$$C = 2 \times 512 \times 3 \times 3 \times 256 \times 26 \times 26 / 10^9$$

$$= 1.595 \text{ BFLOPs}$$

while the replacement of fourth down sampling block with DenseBlock, it yields

$$C = 0.797 + 0.022 + 0.249 + 0.033 + 0.349$$

$$= 1.45 \text{ BFLOPs.}$$

From the above mentioned, the skipping connections between each layer alleviate the insufficient utilization of feature extraction and gradient disappearance caused by cascade structure. Also, the structure has a narrower network width and fewer parameters, so it has faster speed than original model. To improve the YOLOv3 with the parallel structures, the fourth and fifth down samplings are realized by the DenseBlock instead of its original convolution layers with a kernel of \(3 \times 3\) and a stride of \(2\).
Feature pyramid structure

Spatial pyramid pool (SPP) proposes a structure to merge feature vectors of arbitrary size and fixed size. In each spatial bin, SPP pools the input by using the max pooling. The introduced feature pyramid structure and general SPP block are shown in Figure 3. The input feature is $W \times H \times C$. After entering SPPBlock, we get four branches. Three of them use convolution kernels of different sizes, such as $5 \times 5$, $9 \times 9$ and $13 \times 13$. The fourth branch is connected with other branches without any convolution operation. Finally, the network gives an output with a scale of $W \times H \times 4C$.

This method changes the size of the filters in the pooling layer. It partitions the image into divisions from finer to coarser levels, and aggregates local features in them.
Convolution kernels of different sizes have different sizes of reception fields and extract multi-scale features. Spatial features are extracted separately to obtain local information. These features are concatenated to get global features.

5. Improved YOLOv3 model

5.1. The improved network architecture

The improved YOLOv3 network replaces the down-sampling layer scaled of 26 × 26 with the DenseNet structure. As shown in Figure 4, the DenseNet consists of 4 concatenation operations and 8 convolutions. It is composed of a 1 × 1 convolutional layer, a 3 × 3 convolutional layer and a concatenation layer. Four dense blocks of the same structure make up the DenseNet structure. Before the YOLO detector, a SPP module is added. The module contains three maximum pooling, and the last layer of the module connects the results of the three pooling in series. The SPP module can extract features that reflect various sized objects, so it can increase the detection rate for both small and large objects.

After the addition of the DenseNet structure, the multiple layers of features are combined and more feature information is incorporated. It solves the problem of lack of feature information in the feature extraction process. On the other hand, SPP provides convolution kernels of different sizes to enhance the adaption of the model to multi-sized features. Putting the module in different locations on the network will have different effects. Because the deeper the network layer, the richer the semantic information, the more useful the function, and the greater the meaning of information transmission (Lecun et al., 2015). So the module is placed in the deep layer of the network. The feature map $13 \times 13 \times 512$ is fused with shallow features through skip connection. SPP structure is only put before this feature map, for outputting the changed features to each scale.

5.2. Loss function

The loss function is used to evaluate the modelling errors between predicted and real values. In object detection, the total loss value is composed of position loss ($L_{loc}$), classification loss ($L_{cls}$) and confidence loss ($L_{conf}$), i.e.

$$L = \lambda_{coord}L_{loc} - L_{cls} - L_{conf}$$  \hspace{1cm} (5)

$$L_{loc} = \sum_{i=0}^{S^2} \sum_{j=0}^{B} \rho_{ij}^b (2 - w_i \times h_i)((x_i - \hat{x}_i)^2 + (y_i - \hat{y}_i)^2)$$

$$+ \sum_{i=0}^{S^2} \sum_{j=0}^{B} \rho_{ij}^b (2 - w_i \times h_i)((w_i - \hat{w}_i)^2 + (h_i - \hat{h}_i)^2)$$  \hspace{1cm} (6)

where position loss includes coordinate loss and width-height loss; $x_i, y_i, w_i, h_i$ are coordinates of each prediction box; $\hat{x}_i, \hat{y}_i, \hat{w}_i, \hat{h}_i$ are coordinates of each ground-truth box. $\lambda_{coord}$ is a coordination coefficient which is used to coordinate different size rectangles contributing inconsistently to the loss function. An input picture is divided into a $S \times S$ grid. Each grid cell predicts B bounding boxes.

$$L_{loc} = \sum_{i=0}^{S^2} \sum_{j=0}^{B} \rho_{ij}^b \left[ \hat{p}_i(c) \log(p_i(c)) + (1 - \hat{p}_i(c)) \times \log(1 - p_i(c)) \right]$$  \hspace{1cm} (7)

$$L_{cls} = \sum_{i=0}^{S^2} \sum_{j=0}^{B} \rho_{ij}^b \left[ \hat{c}_i \log(c_i) + (1 - \hat{c}_i) \log(1 - c_i) \right]$$

$$- \lambda_{noobj} \sum_{i=0}^{S^2} \sum_{j=0}^{B} \rho_{ij}^{noob} \left[ \hat{c}_i \log(c_i) + (1 - \hat{c}_i) \right] \times \log(1 - c_i)$$  \hspace{1cm} (8)

where $p_i(c)$ and $\hat{p}_i(c)$ are the predicted and ground-true values of the object category. $c_i$ is the probability of an object in the prediction box and $\hat{c}_i$ is ground-true value. $\lambda_{noobj}$ is a weight value, which represents the proportion of the confidence error in the loss function when the prediction box predicts no object, usually set as 0.5.
Figure 4. The improved YOLOv3 model structure.

Table 1. Initialization parameters.

| Size of input images | Batch | Momentum | Initial learning rate | Decay | Training steps |
|----------------------|-------|----------|-----------------------|-------|----------------|
| 416 × 416            | 2     | 0.9      | 0.001                 | 0.0005| 15,000         |

6. Experiments and discussion

The detection models were trained and tested on Ubuntu16.04, Intel® Core i7-8700K 3.70GHz, GeForce GTX1080Ti 11GB, CUDA10.1, cuDNN7.5. The network initialization parameters are shown in Table 1. Training steps were 15,000. The batch size was set to be 2. The model was trained under the training parameters. The learning rate decreased 90% after 10,000 steps and to 0.00001 after 13,000 steps.

6.1. Evaluation indicators selection

The evaluation indicators of the object detection model include recall, precision, Precision-Recall curve (P-R curve), Average-Precision, mAP and so on (Özdemir et al., 2010). There are four concepts. True positives (TP) are the numbers correctly divided into positive (P); False positives (FP) are the numbers wrongly divided into positive; False negatives (FN) are wrongly divided into the numbers of negative examples; True negatives (TN) is the numbers of negative examples correctly divided.

\[
\text{Recall} = \frac{TP}{TP + FN} = \frac{TP}{P} \quad (9)
\]

\[
\text{Precision} = \frac{TP}{TP + FP} \quad (10)
\]

A certain confidence level was set as a threshold. When it is greater than this threshold, a P-R curve can be drawn for a certain class. The area enclosed by this curve and the coordinate axis is the average accuracy of the class. When there are \( N \) classes of objects, mAP is the average of the sum of the average precision of all classes.

\[
mAP = \frac{\sum_{i=1}^{N} AP_i}{N} \quad (11)
\]

6.2. Image data preparation

The dataset contained 665 pictures of people wearing helmet. Since a small amount of training data caused overfitting in the deep network, the existing dataset was expanded by increasing the number of images in the dataset. In order to enhance the generalization ability and model robustness, the collected images were preprocessed by data enhancement methods including rotation, translation, scaling, random occlusion, horizontal flip, and noise addition. The main methods used here are jitter, hue, saturation, exposure, and flip.

In order to compare the performance of different algorithms, images in the training set were marked to PASCAL VOC format. After using the script to number the image, we used the label tool labelImg to tag the image to get the XML file. If positive samples were too small or blurred, we did not label them to prevent over-fitting in the neural network.

Nine groups of anchor box defined in the original YOLOv3 network are calculated by the COCO dataset. In order to obtain anchors more suitable for this dataset, we used \( K \)-means algorithm to cluster the dataset. The newly obtained 9 sets of anchors were (6,11), (10,16), (13,22), (17,28), (22,37), (29,49), (40,68), (58,101) and (101,170) respectively. High-resolution detection used the small
Figure 5. Loss curve (left) and P-R curve (right).

Figure 6. Comparison of model results. a. The original network (wear a helmet), b. The improved network (wear a helmet), c. The original network (no wear a helmet), d. The improved network (no wear a helmet), e. The original network (both), f. The improved network (both).
sized frame, while the low-resolution detection of large objects used the large sized.

### 6.3. Improved YOLOv3 detection results

We used the improved YOLOv3 to detect candidate helmet regions for the input images. The loss curve and P-R curve obtained after using the improved network training is shown in the Figure 5. It shows that the loss value can quickly converge in a small number of iterations, which depends on the weight sharing of transfer learning in the shallow convolutional layer. In addition, if there is a larger accelerated graphics card, the model convergence and detection speed will be further improved.

Figure 6 shows the comparison results between the improved YOLOv3 network and the original YOLOv3 network. The pictures at the left column are the detection results of the original YOLOv3 while the three pictures at the right column are the improved YOLOv3 model results. Where Figure 6(a) predicted an incorrect label different from the Figure 6(d). Also, Figure 6(e) missed an object detection compared to Figure 6(f).

In order to verify the high performance of this model, the dataset was trained and tested on the four different algorithms, i.e. the YOLOv3 original network, SSD, Faster R-CNN and improved YOLOv3. The mAP between these algorithms is shown in Table 2. Experiments showed that the improved YOLOv3 had the highest detection accuracy for helmets.

### 7. Conclusions

This paper proposed an improved YOLOv3 algorithm for object detection. Firstly, a dataset was created by a web crawler and annotated it, and then the dataset was clustered to optimize the anchor parameters. Due to DenseNet structure and SPPNet structure were introduced, deep features fused with shallow features and network accuracy improved. Finally, the multi-objective loss function combined with mean square error loss and cross entropy loss was used to regress and correct the prediction frame, so the accuracy of network detection was improved. Experimental results show that the improved YOLOv3 algorithm can effectively improve the detection accuracy of the helmet. Because this paper presents only a theoretical model, the next step is to deploy on the embedded system for research.

### Acknowledgements

This research was partially supported by the Innovation Scientists and Technicians Troop Construction Projects of Henan Province (grant number CXTD2016054), Zhongyuan high level talents special support plan (grant number ZYQR201912031), Key Scientific Research Projects of University in Henan Province (grant number 21A120004) and the Fundamental Research Funds for the Universities of Henan Province (grant number NSFRF170501).

### Disclosure statement

No potential conflict of interest was reported by the author(s).

### Funding

This work was supported by Innovation Scientists and Technicians Troop Construction Projects of Henan Province: [grant number CXTD2016054]; Fundamental Research Funds for the University of Henan Province: [grant number NSFRF170501]; Key Scientific Research Projects of Universities in Henan Province: [grant number 21A120004]; Zhongyuan high level talents special support plan: [grant number ZYQR201912031].

### References

Girshick, R. (2015). Fast R-CNN. In *IEEE International Conference on Computer Vision (ICCV'15)*, 1440–1448. [https://doi.org/10.1109/ICCV.2015.169](https://doi.org/10.1109/ICCV.2015.169)

He, K., Zhang, X., Ren, S., & Sun, J. (2015). Spatial Pyramid Pooling in Deep Convolutional Networks for Visual Recognition. *IEEE Transactions on Pattern Analysis & Machine Intelligence*, 37(9), 1904–1916. [https://doi.org/10.1109/TPAMI.2015.2389824](https://doi.org/10.1109/TPAMI.2015.2389824)

He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep residual learning for image recognition. *IEEE Conference on Computer Vision & Pattern Recognition(CVPR’16)*, 770–778. [https://doi.org/10.1109/CVPR.2016.90](https://doi.org/10.1109/CVPR.2016.90)

Huang, G., Liu, Z., Van Der Maaten, L., & Weinberger, K. Q. (2017). Densely connected convolutional networks. *IEEE Conference on Computer Vision and Pattern Recognition(CVPR'17)*, 2261–2269. [https://doi.org/10.1109/CVPR.2017.243](https://doi.org/10.1109/CVPR.2017.243)

Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2012). ImageNet classification with deep convolutional neural networks. *Advances in Neural Information Processing Systems*, 1097–1105. [https://doi.org/10.1145/3065386](https://doi.org/10.1145/3065386)

Lecun, Y., Bengio, Y., & Hinton, G. (2015). Deep learning. *Nature*, 521(7553), 436–444. [https://doi.org/10.1038/nature14539](https://doi.org/10.1038/nature14539)

Li, F. F., Perona, P., & Technology, C. I. O. (2005). A Bayesian hierarchical model for learning natural scene categories. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR’05)*, 2, 524–531. [https://doi.org/10.1109/CVPR.2005.16](https://doi.org/10.1109/CVPR.2005.16)

Lin, T., Dollár, P., Girshick, R., He, K., Hariharan, B., & Belongie, S. (2017). Feature Pyramid Networks for Object Detection. *IEEE Conference on Computer Vision and Pattern Recognition(CVPR'17)*, 936–944. [https://doi.org/10.1109/CVPR.2017.106](https://doi.org/10.1109/CVPR.2017.106)

### Table 2. The mAP between common algorithms.

|        | SSD with VGG16 net | Faster R-CNN | Improved YOLOv3 |
|--------|--------------------|--------------|-----------------|
| AP (Hat) | 0.764              | 0.871        | 0.848           | 0.851           |
| API (Person) | 0.661             | 0.792        | 0.853           | 0.922           |
| mAP    | 0.709              | 0.831        | 0.851           | 0.886           |
Liu, W., Anguelov, D., Erhan, D., Szegedy, C., Reed, S., Fu, C., & Berg, A. C. (2016). SSD: Single Shot MultiBox Detector (pp. 21–37). European Conference on Computer Vision.

Liu, H., Cao, F., Wen, C., & Zhang, Q. (2020). Lightweight multi-scale residual networks with attention for image super-resolution. *Knowledge-Based Systems*, 203. https://doi.org/10.1016/j.knosys.2020.106103

Özdemir, B., Aksoy, S., Eckert, S., Pesaresi, M., & Ehrlich, D. (2010). Performance measures for object detection evaluation. *Pattern Recognition Letters*, 31(10), 1128–1137. https://doi.org/10.1016/j.patrec.2009.10.016

Redmon, J., Divvala, S., & Girshick, R. (2016). You Only Look Once: Unified, Real-Time Object Detection. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR’16)*, 779–788. https://doi.org/10.1109/CVPR.2016.91

Redmon, J., & Farhadi, A. (2017). YOLO9000: Better, faster, stronger. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR’17)*, 6517–6525. https://doi.org/10.1109/CVPR.2017.690

Redmon, J., & Farhadi, A. (2018). YOLOv3: An incremental improvement. *arXiv preprint*. https://arxiv.org/abs/1804.02767

Ren, S., He, K., Girshick, R., & Sun, J. (2017). Faster R-CNN: Towards real-time object detection with region proposal networks. *Advances in Neural Information Processing Systems*, 39, 1137–1149. https://doi.org/10.1109/TPAMI.2016.2577031

Tian, Y., Yang, G., Wang, Z., Wang, H., Li, E., & Liang, Z. (2019). Apple detection during different growth stages in orchards using the improved YOLO-V3 model (pp. 157). Elsevier B.V. https://doi.org/10.1016/j.compag.2019.01.012

Zhou, P., Ni, B., Geng, C., Hu, J., & Xu, Y. (2018). Scale-transferrable object detection. *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR’18)*, 528–537. https://doi.org/10.1109/CVPR.2018.00062